SPATIAL COMPUTING OF SOUND FIELDS IN VIRTUAL ENVIRONMENTS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Zechen Zhang

August 2019

SPATIAL COMPUTING OF SOUND FIELDS IN VIRTUAL ENVIRONMENTS

Zechen Zhang, Ph.D.

Cornell University 2019

Over the past decades, computer graphics researchers have put enormous effort into rendering realistic visual scenes by simulating light transport. With the high-level goal of creating realistic immersive experiences in virtual worlds, physically plausible sound is a critical piece and remains to be explored.

Humans' experience when perceiving sound is spatially varying and scene dependent, e.g. whether a point sound source is occluded or not with respect to listener will lead to a different perceived sound. Simulating sound propagation is the key to reproducing such effects but it differs from light transport simulation in visual rendering due to the importance of diffraction effects. By using physical simulations, the grand goal of this thesis is to provide auditory cues that respect the influence of the virtual environment.

We address this problem by precomputing an expensive simulation of sound wave propagation through a voxelized 3D scene and encoding perceptually important acoustic parameters per voxel from the simulation data, which enables efficient real-time sound rendering at run-time. All methods proposed are immediately practical with potential applications in AR/VR and gaming.

Our first contribution is proposing a framework that simulates ambient sound propagation in a preprocessing stage and reconstructs ambient sound efficiently at render time. By modelling spatio-temporally incoherent ambient sound source appropriately in numerical simulation, a streaming encoder captures the **loudness** and **directivity** per listener posi-

tion compactly in spherical harmonics coefficients. The encoded coefficients are further coupled with Head-related transfer function (HRTF) data, rendering physically plausible binaural ambient sound at run-time.

We then observe that in most ambient sound scenarios, the **sound texture** perceived varies in space. For example, near a water stream, crisp water bubble sounds are audible with transient details, whereas it becomes closer to a randomized colored noise in far field. A more compelling example is a babbling crowd, in which individual speech is recognizable next to a person but not when far away from the crowd. The intuition is that the perceived ambient sound is a random collection of similar micro sound events and the variation of atomic sound events' temporal density and their distribution of amplitudes leads to different sound textures. We propose a simple ambient sound texture representation in terms of an event density function (EDF). By modelling micro sound events directly in the precomputed simulation phase, EDF is compactly encoded. At run-time, sound is rendered by real-time granular synthesis, resulting in a spatially varying sound texture that enhances the experience of ambient sound in a virtual environment.

BIOGRAPHICAL SKETCH

Zechen Zhang was born on January 12th, 1991 in Cixi, Zhejiang, China. He did his undergraduate studies at University of Science and Technology of China (USTC) and obtained a B.S. degree in Applied Physics in 2013. He joined the Ph.D program in School of Electrical and Computer Engineering at Cornell University subsequently. He did graduate studies in the area of computer graphics and received his doctorate degree in 2019.

For Yuguang

ACKNOWLEDGEMENTS

I would like to thank all my friends and colleagues who helped and supported me all the way through.

First, I want to express my deepest gratitude to my advisor Professor Steve Marschner, who guided me through my long journey exploring computer graphics as a graduate student. Over the years, he showed me how to find, approach and solve a scientific problem and I can never over emphasize how much I learned from his remarkable instinct on filtering out the essence of a complex problem. I also would like to thank Steve for always being supportive and allowing me to work on topics outside traditional graphics rendering area.

I enjoyed my three internships in Interactive Media Group (IMG) at Microsoft Research Labs, Redmond. I would like to thank John Snyder and Nikunj Raghuvanshi sincerely for kindly serving as my intern mentors and collaborating with us over the past years. John's innovative ideas are always great sources of inspirations and his research attitude of doing the simple and 'right' thing is something I have been aspiring. Nikunj introduced me into computational acoustics field and taught me so many things on acoustics, signal processing and simulation.

I thank Professor Anil Damle and Professor Al Molnar for serving as my committee members and reading this dissertation. The feedbacks Anil and Al gave were very constructive and beneficial for me.

I also owe thanks to all my friends in Cornell graphics and vision group, MSR, ECE department, FICCC and LECC. They offered me so much encouragement and prayers during my lonely and tough years. This dissertation would never have been possible without

them.

I am deeply obliged to my parents, Jingrong and Fengxian, who have been giving me coutless unconditional love and support. Their guidance when I was a child and willingness of allowing me to follow my passion definitely shaped who I am today.

Finally, I would like to thank my wife, Yuguang, for her endless love and support. Without her, I cannot imagine how my life in Ithaca would look like. This dissertation is dedicated to her.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In recent years, image synthesis techniques in computer graphics have achieved huge success in rendering a stunning photorealistic virtual world. Yet, within the high level goal of providing an immersive user experience, sound, as another important piece of human perception, remains less widely studied. On the other hand, the audio research community has proposed numerous 3D spatial sound methods for rendering sound from headphones or loudspeaker-arrays to control multiple perceptually relevant sound effects [5], e.g. the direction of sound arrival, loudness and reflection and reverberation properties. However, making spatial sound rendering techniques aware of scenes remains a hard challenge because it requires simulating a complex sound transport process in 3D space. This makes the perceived sound inconsistent with visual cues, degrading the overall immersive experience.

This thesis attacks several open problems in spatial sound simulation and rendering with the goal of *making the rendered sound respect the spatial 3D geometry of the scene.*

In both hearing and vision, human perceptual systems discount environment influences to better identify the objects' characteristics. However, putting these other influences back in provides useful cues for sensing the environments. In vision, illumination conditions change under different environments; in hearing, reverberation is affected by the size of the surrounding room size. An enormous amount of work has been done in the computer graphics field to draw 3D shaded objects with shadows and global illumination effects. On

the other hand, most existing spatial sound works render sound entities only with specified directions and sizes, without considering environment effects. Integrating environment cues into spatial sound rendering provides the user a more realistic sound result which also aligns with perceived visual information.

Physically rendering a visual scene boils down to simulating light transport in 3D space. Similarly, scene-aware spatial sound rendering requires sound transport simulation, whose inputs include 3D geometry with tagged materials and sound source / listener properties.

In general, there are two approaches to simulate the sound transport process. *Geometric acoustics* methods use a small-wavelength approximation, ignoring diffraction effects and adopting path-tracing like techniques. They are able to simulate sound propagation from point sources interactively and to support dynamic geometry. However, sound waves have much larger wavelengths than light waves and exhibit important diffraction effects which can not be modeled by geometric methods directly. Geometric acoustics loses physical accuracy in the near-field and at low frequencies where diffraction effects are dominant. Further, for volumetric sound sources, the convergence rate of path tracing methods prohibits interactive simulation and rendering. The other approach to sound transport simulation is *physical wave simulation*. It solves the acoustic wave equation directly, which automatically captures all physical cues, including diffraction. However, its computational cost is much higher than path-tracing methods in a simple scene, so it cannot be used for real-time simulation of sound propagation in 3D space. One way to circumvent the high cost while retaining physical accuracy is to precompute the wave simulation offline before

rendering sound in real-time.

An important sound phenomenon we studied extensively is *ambient sound*. Ambient sounds are common in physical environments. They usually originate from static extended sound sources rather than from point sources and are composed of many spectrally similar micro sound events. We seek to efficiently capture and render salient propagation effects to improve the realism of ambiences in games and virtual reality. Most prior spatial sound simulation and rendering methods aim at point-like sound sources. However, modeling these events individually is impractical in interactive applications which typically support about a hundred active sounds in total and allocate only a few for the background. Replacing this complexity by a few point proxies causes unrealistic wobbles in loudness as the listener moves past the proxies and fails to reproduce the correct aggregate effects of distance falloff and shadow softening. All these factors make scene-aware ambient sound synthesis challenging.

This thesis uses a precomputed physical wave simulation paradigm to address static ambient sound simulation and rendering. Given a voxelized 3D scene, an expensive sound propagation simulation is conducted in a precomputation phase and compact perceptual acoustic information is extracted at potential listener 3D positions, which is then further compressed over space. At run time, the spatial sound effects are rendered in real-time using the precomputed perceptual parameters at the dynamic listener position. This paradigm physically simulates sound transport and is still able to render spatial sound in real-time at

dynamic listener positions.

1.1 Overview of contributions

Most of the methods in this dissertation are built upon the parametric wave-field simulation and encoding framework proposed in [30]. Our contributions are as follows:

Ambient sound propagation We propose a first-of-its-kind system addressing realistic 3D spatial sound effects from ambient sound sources in virtual environments, e.g. rain and waterfalls. By precomputing sound transport simulation from spatio-temporally incoherent volumetric sound sources, the system automatically captures the **loudness** and **directionality** cues from ambient sound, without any human effort.

Acoustic texture simulation and rendering Besides the first order **loudness** and **directionality** cues, this work also simulates and renders **acoustic texture** variation in 3D space, capturing the indistinct murmur of a faraway brook versus the bright babbling of one up close. The key contribution is formalizing the notion of acoustic texture variation by introducing the *event density function (EDF)*, which relates the rapidity of received events as a function of their individual loudness and listener location in the scene. In a single precomputed wave simulation, the ambient sound source is modeled as a collection of impulsive sound events and the EDF is encoded. At render time, our system uses real-time granular synthesis to render ambient sound according to a dynamic EDF, producing plausible texture variations as the listener moves around in complex virtual scenes.

1.2 Orgnization of the thesis

This thesis is organized as follows: chapter 2 reviews related previous work and clarifies our problem statement; chapter 3 explicitly states the numerical methods used in our precomputed finite-difference time-domain (FDTD) simulation; chapter 4 proposes an ambient sound simulation framework that uses a spatio-temporal incoherence idealization and enables lightweight rendering method for loudness and directivity cues; chapter 5 proposes a new framework that simulates and renders the acoustic texture efficiently; chapter 6 contains the conclusions and also looks forward for future work.

CHAPTER 2

BACKGROUND

This dissertation attacks an open problem in audio-visual computing: how to physically render generic ambient sound in 3D space. In the following chapters, we propose two systems that reproduce realistic ambient sound in virtual environments.

2.1 Problem statement

Ambient sounds like wind, rain, or surf provide a dynamic background, propagating through a 3D scene to complement visuals and immerse the listener in an environment. Ambient sources superpose many independent, spectrally similar atomic *source events* overlapping in time and distributed over the source's spatial extent, such as individual rain drops, oscillating bubbles in a stream, or bird tweets in a flock. These events occupy a similar frequency band within human auditory perception and we perceive them as an aggregate.

As the listener navigates, multiple aspects of ambient sound are important cues for human ambient sound perception.

- *Loudness* changes convey the size and shape of the sound source. For instance sound attenuates when moving away from the beach but not when moving along it. Variation due to scene occlusion indicates how open or enclosed the surrounding space is.

- *Directionality* from sound streaming through portals or adjoining chambers reveals their presence even when behind the listener. The beach sounds big outside but becomes directionally crisp when heard through an open door or window.
- *Acoustic texture*, which arises from audible statistical properties of atomic source events, is spatially varying. For instance, a faraway stream sounds noise-like but becomes a more distinct babbling with individually recognizable drips and gurgles as one gets closer. The texture varies not only with distance but also from sound propagation within the scene: rain sounds different when heard indoors through a door compared to outside, not just because it gets fainter, but also because events get mixed at the door and are multiplied via reverberation within the room.

2.2 Sound transport simulation approaches

To simulate sound transport in 3D environments, there are two types of existing simulation approaches: geometric acoustics (GA) and physical wave acoustics simulation methods.

2.2.1 Geometric acoustics

Geometric acoustics (GA) methods were first investigated in ray tracing [21] and image source [2] methods. They are surveyed in [37] and most of them amount to a small-wavelength approximation to the wave equation. GA techniques are able to simulate sound propagation at interactive rates and support dynamic geometry, at the cost of sacrificing

physical accuracy: they use a limited number of sampled paths and must approximate physical diffraction effects.

GA methods require transport path sampling to leverge ray tracing techniques. Traditionally only low-order reflection paths are sampled for real-time sound propagation simulation due to computational cost. Recently, by neglecting diffraction, bi-directional path tracing has shown promise [9] for point sound sources. However, one important phenomenon this thesis studies is ambient sound from an extended source. Typical extended sources that we consider for ambient sound sources comprise thousands of source events; it becomes very costly for GA to obtain converged estimates that find all paths connecting each source event to the listener. [39] renders large volumetric sound sources in free field conditions, but the question of how to render volumetric sources in a physical way in the presence of geometry remains to be answered.

How to include diffraction in a physically correct fashion is a more fundamental problem for GA methods. The Biot−Tolstoy−Medwin (BTM) diffraction model [4, 25, 8] has been widely used to account for the prominent role of diffraction in audible-wavelength sound propagation. BTM models the diffracted field around an edge as a superposition of wavelets emitted from elementary parts of the edge. In simulation, the edge needs to be discretized, making the diffracted field a summation of wavelets from all edge elements. For high-order diffraction, all possible combinations of elements on each edge need to be included, yielding an exponentially growing complexity. Recent works use stochastic scattering at or near geometric edges where sound paths bend and techniques are being investigated in [38]. But the general problem remains open [37].

2.2.2 Physical wave acoustics

Solving the wave equation directly captures diffraction [14] and explores all these paths systematically but implicitly, but at a computational cost that prohibits interactive movement of sources and listener. Previous systems [34, 30] precompute the simulation and extract compact perceptual acoustic information (e.g. arrival delay, loudness, energy decay rate, etc.), enabling real-time rendering of propagation effects. Multiple stunning scene-aware sound effects can be rendered in real-time systems, such as echoes in a reverberant chamber [30] and directions of the sound arrival [32]. The simulations are physical and the rendered sounds are usually more plausible compared with GA methods.

Most previous wave-based techniques consider a single point source; our interest lies in large ambient sources comprising thousands of independent source events. Our parameterization of the aggregate acoustic response also differs. [30] extract loudness for the direct sound, early reflections, and late reverberation transient phases of the impulse response for rendering perceptually important acoustic effects on sound emitted from a single coherent point source. Our work focuses on capturing the variable *loudness*, *directionality* and *statistical texture* of ambient sounds and the modification of the properties by sound propagation.

2.3 Sound texture synthesis

Perceptual sound texture analysis and synthesis Research in sound perception has been studying the problem of generating a new sound texture that is perceptually similar to a given audio clip, but not identical to the input (nor to trivial transformations of the original signal, like translation). Many existing algorithms in the field are inspired by visual texture synthesis architectures. [23, 24] proposed statistical descriptions that can be extracted from recorded sounds to summarize the perceived sound texture. These methods focus on stochastic temporal fluctuations in acoustic energy across different audio frequency bands, and they can be used to transform Gaussian random noise into sounds matching the original. Recent advances also use neural networks to synthesize audio textures with improved diversity and quality [3].

We are interested in a related but different problem: characterizing and rendering the spatially-varying sound texture due to propagation within a complex scene. We propose a statistical representation of sound texture [54] called the event density function (EDF), which is independent of the particular sounds emitted by the source. This modification is what we term acoustic texture, captured in the EDF. Once it has been extracted for a source volume and scene, our run-time rendering system lets the user specify any grain sounds, synthesizing the resulting radiated sound and applying the acoustic texture modification in real-time.

Granular sound synthesis Granular synthesis is a broad term encompassing many techniques in audio processing and synthesis that break a sound into short (10–100 ms) units called *grains* and then reassemble them by concatenation or blending to modify (e.g., pitch shift) the original sound or to synthesize new sounds [13, 36, 51]. Specialized techniques for rendering particular sounds such as rain have been proposed [27]. Our run-time rendering employs granular synthesis, where grains are complete individual source event sounds which are generated and mixed stochastically to render the output. Our focus is on informing this process with the virtual environment.

CHAPTER 3

FDTD WAVE SIMULATION BASICS

Sound can be interpreted as air particle vibrations and propagates as an audible wave of pressure through media. *Acoustic pressure* is the local deviation of pressure from equilibrium. It is a scalar-valued function of both 3D space and time. The goal of a time-domain acoustic simulator is reconstructing the acoustic pressure variation in space-time space.

The input of our simulation pipeline is the 3D scene geometry with tagged material properties, listener positions, sound source volume and sound source signal. The transport of sound waves in air is governed by the scalar acoustic wave equation

$$\frac{1}{c^2}\frac{\partial^2 p}{\partial t^2}(x,t) - \nabla^2 p(x,t) = s(x,t) \tag{3.1}$$

where x is 3D spatial location, t is time, c=340 m/s is the speed of sound, and p is scalar acoustic pressure to be solved for. The (input) source term $s(x,t)$ represents the source perturbation to be propagated and will be detailed in chapter 4 and 5. Note that by solving equation 3.1 numerically, complex wave effects such as diffraction and interference are included by default.

There exist various methods solving wave equation 3.1 in time domain. In our precomputation phase, the finite difference time domain (FDTD) numerical method [47] is used to simulate wave propagation. It was originally proposed for vector-valued electromagnetic wave propagation simulation [52] and is well suited for solving the scalar acoustic wave equation 3.1 as well.

FDTD is simple to implement. Its major disadvantages are numerical dispersion, causing different frequencies to incorrectly travel at different speeds, and numerical dissipation, causing high frequencies to be attenuated. These limitations will be discussed in section 3.3.

3.1 Standard FDTD scheme

The FDTD method solves equation 3.1 on regular 3D voxels at discretized time steps. Let $p^i_{l,m,n}$ denote the pressure field at the i-th time step over voxels indexed (l, m, n). Applying second order central finite difference for both space and time domain derivatives, equation 3.1 is discretized as

$$p^{i+1}_{l,m,n} = \lambda^2(p^i_{l+1,m,n} + p^i_{l-1,m,n} + p^i_{l,m+1,n} + p^i_{l,m-1,n} + p^i_{l,m,n+1} + p^i_{l,m,n-1}) - p^{i-1}_{l,m,n} + c^2\Delta t^2 s^i_{l,m,n} \tag{3.2}$$

where the Courant number $\lambda = \frac{c\Delta t}{\Delta x}$, $c = 340$ m/s is sound wave speed, Δt is the discrete time step, and Δx is the regular 3D voxel size.

3.2 Acoustic power flux

Flux density (or simply, flux) represents instantaneous power transport in the fluid across a differential oriented area, analogous to vector irradiance in optics. It estimates the direction of a wavefront passing x at time t, via

$$f(x,t) = p(x,t)\, v(x,t), \quad v(x,t) = -\frac{1}{\rho_0}\int_{-\infty}^{t} \nabla p(x,\tau)\, d\tau \tag{3.3}$$

where v is the particle velocity and ρ_0 is the mean air density (1.225kg/m^3). We compute flux density on each voxel as the simulation runs and this variable is dedicated to directivity-aware sound spatialization which will be discussed in following chapters.

Central finite differences among 3D voxels are used to compute spatial derivatives for ∇p, and the midpoint rule for numerical time integration, yielding the update equation for x component of particle velocity as

$$(v_x)^{i+1}_{l,m,n} = (v_x)^{i}_{l,m,n} - \frac{\Delta t}{\rho_0} \frac{p^i_{l+1,m,n} - p^i_{l-1,m,n}}{\Delta x} \tag{3.4}$$

And similarly for its y and z components.

3.3 Numerical limitations

Numerical dispersion and dissipation form the main limits on the highest frequency we can simulate for a given grid spacing.

With specified grid spacing Δx, the time step is

$$\Delta t = \frac{\lambda \Delta x}{c}, \tag{3.5}$$

where the Courant number λ is at most $1/\sqrt{3}$ ensuring CFL stability [10] in 3D simulation, and the Nyquist frequency is

$$f_n = \frac{1}{2\,\Delta t}. \tag{3.6}$$

A simulation with duration T requires N_t time steps

$$N_t = \frac{T}{\Delta t}. \tag{3.7}$$

Simulation cost is proportional to the product of total number of voxels and number of time steps N_t. Assuming n voxels along each spatial dimension and the simulated duration is fixed, the time complexity is $O(n^4)$ while the memory complexity is $O(n^3)$.

We use standard leap-frog integration as discussed in section 3.1, in which dissipation is greatest along axial directions [40, 19] with cut-off frequency

$$f_d = \frac{1}{\pi \Delta t} \sin^{-1}(\lambda) \tag{3.8}$$

To avoid unwanted numerical dissipation at high frequencies, the injected source signal $s(x, t)$ does not contain significant content beyond axial cutoff frequency f_d.

Low frequencies in the source signal should also be removed: a non-zero DC component generates a non-zero particle velocity (i.e. wind) in the direction away from the source. We therefore limit the power spectral density (PSD) content of the source signal to $[f_{\min}, f_{\max}] \subset (0, f_d]$ and denote its bandwidth $\Delta f = f_{\max} - f_{\min}$.

Numerical dispersion is another FDTD artifact whose effect is frequency dependent. Travelling waves with frequency closer to the dissipation limit f_d tend to disperse more than lower frequencies. To avoid this while balancing computational cost, a rule of thumb is to set the ratio between smallest wavelength injected and voxel size Δx around $4 \sim 5$ in practice. Simulated source signal $s(x, t)$ will be detailed in chapter 4 and chapter 5 separately.

Our simulation is computed on a cuboid domain. The simulation duration is denoted $T = T_S + T_D$, where T_S is the length of time in which the source continues to emit event pulses and T_D is the time required for a wave to propagate across the diagonal of the

simulation domain. The fixed value $T_S = 2$s builds reliable and spatially-smooth statistics for the studied acoustic parameters in all our experiments.

3.4 Boundary conditions

We use a Neumann boundary condition [49] to deal with wave-geometry interactions at interfaces. It enforces that the gradient of scalar acoustic wave pressure w.r.t surface normal is always zero, implying even symmetry of the pressure field about the interface. Intuitively, Neumann boundary conditions specularly reflect waves and the reflection density function is frequency independent. This assumption makes sense considering the long wavelength of acoustic waves compared with geometry features. Frequency dependent surface reflection modelling is still an open problem [20] and we will leave it for future work.

3.4.1 Perfectly matched layer

The simulation domain is truncated to balance computation cost and physical accuracy. It is intended to model sound emission into a free space without any back reflection. However, simple boundary conditions typically trigger reflections for incident waves. To suppress spurious reflections from the domain boundary, a perfectly matched layer (PML) absorber is placed on each domain boundary face. It uses stretched coordinate variables and models wave propagation in an unphysical lossy medium. The impedance of the lossy

medium matches with the air at the media-air interface, preventing reflection errors. More details can be found in [35].

We also couple the PML with the specularly reflecting walls in the virtual environment, augmenting the reflecting boundary condition with a variable reflectivity [29]. When updating the scalar wave pressure field at the geometry interfaces at one time step, the Neumann and PML boundary conditions are evaluated separately, resulting in two different wave fields for the following time step. The updated wave field value is a weighted summation of the results from the two boundary conditions, where the weights are decided by the wall absorption coefficient.

CHAPTER 4

AMBIENT SOUND PROPAGATION

Ambient sounds provide a dynamic background, propagating through a 3D scene to complement visuals and immerse the listener in an environment. As the listener navigates, *loudness* changes convey the size and shape of the sound source. For instance sound attenuates when moving away from the beach but not when moving along it. Variation due to scene occlusion indicates how open or enclosed the surrounding space is. *Directionality* from sound streaming through portals or adjoining chambers reveals their presence even when behind the listener. The beach sounds big outside but becomes directionally crisp when heard through an open door or window. In this work, we capture and render both *loudness* and *directionality* from ambient sound sources for the first time, within a small runtime budget appropriate for background sounds in games and virtual reality (VR).

We assume ambient sources superpose many independent, point-like elementary sound events, such as the impact of each drop in a rain shower. When the listener is unable to distinguish these individual sound events, we observe that ambient sources can be idealized as incoherent in space and time. This idealization fails for individual conversations heard at a crowded party or for nearby cars on a street corner, but it suffices as the crowd's chatter or highway noises merge at greater distance. Prior work on propagation from extended sources has instead focused on spatially coherent sources at most a few meters across, such as free-field radiation from vibrating shells [17] and scattering/shadowing from discrete objects in sparse outdoor scenes [26]. Coherent wave fields are highly oscillatory over space, making these techniques compute- and memory-intensive. The (time-averaged)

power distribution of an incoherent field is smooth, saving CPU and RAM.

Given the source's spatial power distribution, we precompute a single 3D wave simulation using the finite-difference time-domain (FDTD) method. This Eulerian approach naturally accounts for diffraction and scattering, and it propagates the entire field at once, making precomputation time insensitive to the source's spatial extent. In this chapter, we propose the notion of an incoherent wave source and an efficient formulation that generates phase-decorrelated, bandlimited noise signals, yielding response fields with smooth power variation, avoiding spatial oscillations from interference. The simulation output is a pressure field over time and space; it captures how sound propagates from the source but is too large to store in raw form and does not make the information needed at runtime readily available. A streaming encoder is used that efficiently extracts and stores compact perceptual information for incoherent fields. We compute a 3D grid of low-order spherical harmonic (SH) coefficients representing the directional distribution of acoustic power at points regularly sampled throughout the scene. Our encoder computes the flux density vector at each timestep and sample point, and incrementally accumulates the SH coefficients. Prior techniques [22, 33] also apply flux but on short, coherent signals for which overlapping arrivals in different directions cannot be discriminated. With a sustained, incoherent source signal, we show for the first time that flux can tease apart the steady-state spherical power distribution.

We also show that incoherence can be exploited for faster binaural rendering at runtime. Existing techniques require expensive convolution with the HRTF (head related transfer function) to preserve interaural phase differences. But in an ideally incoherent

sound field, phase relationships are difficult to sense and frequency-dependent interaural loudness differences dominate. These can be achieved cheaply using standard parametric equalization. We obtain good results with just four frequency bands, by equalizing the mono-aural representative sound to be propagated using 4-channel left/right gains computed for the current listener position and head orientation in the scene.

Overall, this is the first system to model salient directional propagation effects from arbitrarily large ambient sources in complex game environments. Precomputation takes about several hours on a single desktop. Runtime cost per extended source is less than 1MB of memory and about 100μs per-frame computation on a single core. Cost is largely insensitive to source size or scene's geometric complexity. Our technique integrates transparently with standard game engines like Unreal Engine 4.

4.1 System overview

The designer tags surfaces or volumes in a 3D scene as the extended source. Our goal is to play back in real time a (usually pre-recorded) *representative sound* as if it were emanating from that source and propagating through that scene. Loudness and directionality cues should change in a smooth and natural way as the listener moves.

Our system works in two stages. In the precomputation stage, we voxelize a 3D domain containing the extended source and scene geometry and introduce sustained, volumetric noise. This noise signal is entirely synthetic and used to predict sound transport effects, with no relationship to the representative sound played at run-time.

A wave simulation then propagates this noise throughout the scene. Parameter fields are encoded representing the time-averaged power distribution as a (5D) function of position and direction at the listener, where directionality is represented using low-order (n=4) SH. These fields then modify the representative sound clip at playback time given the listener's changing position and orientation.

The emitted power distribution, shape, and location of the extended source is baked in with respect to the static scene and can't be changed at runtime. Any input clip can be used at runtime to represent the source sound; it should be suitably "ambient" with loudness and spectral content not changing too abruptly.

4.2 Precomputed simulation

As mentioned in chapter 3, we use the FDTD numerical method [47] to simulate wave propagation by solving the wave equation 3.1. Its (input) forcing term $s(x,t)$ represents the source perturbation to be propagated and will be detailed in Section 4.3.

FDTD has numerical dispersion and dissipation errors. As discussed later in Section 4.4, the acoustic parameters we're interested in depend on the response signal's time-averaged power and neglect its phase, making them insensitive to dispersion. Dissipation thus forms the main limit on the highest frequency we can simulate for a given grid spacing.

Simulation cost is proportional to the product of scene volume, spatial resolution, tem-

poral resolution, and duration. Output quality depends on resolution, which determines the field's spatial detail, and the number of frequency bins resolvable in the band where the source propagates without artifacts, which determines the amount of randomness we can inject. We fix the spatial grid spacing (Δx) and number of frequency bins (ΔN) to achieve the required quality, and determine the other parameters as follows.

As discussed in chapter 3, N_t time steps are included in the time domain simulation. The simulated pressure response at any grid point thus yields a corresponding number of "bins" or frequency samples in its minimal discrete Fourier transform given by

$$N = \lfloor N_t/2 \rfloor + 1. \tag{4.1}$$

To suppress dissipation and non-zero particle velocity (chapter 3), we limit the power spectral density (PSD) content of the source signal to $[f_{\min}, f_{\max}] \subset (0, f_d]$ and denote its bandwidth $\Delta f = f_{\max} - f_{\min}$.

The number of frequency bins included in this band is then

$$\Delta N = N \frac{\Delta f}{f_n}. \tag{4.2}$$

We'll see in the next section that the bigger ΔN, the more the simulation approaches ideal incoherence.

We set $\Delta x = 0.255$m, yielding the sampling rate $1/\Delta t = 2309$Hz and dissipation cutoff $f_d = 452$Hz. We set the source PSD's non-zero band as $[f_{\min}, f_{\max}] = [62.5, 400]$Hz and the number of frequency bins included in it as $\Delta N = 1000$. This yields $N_t \approx 9200$ in

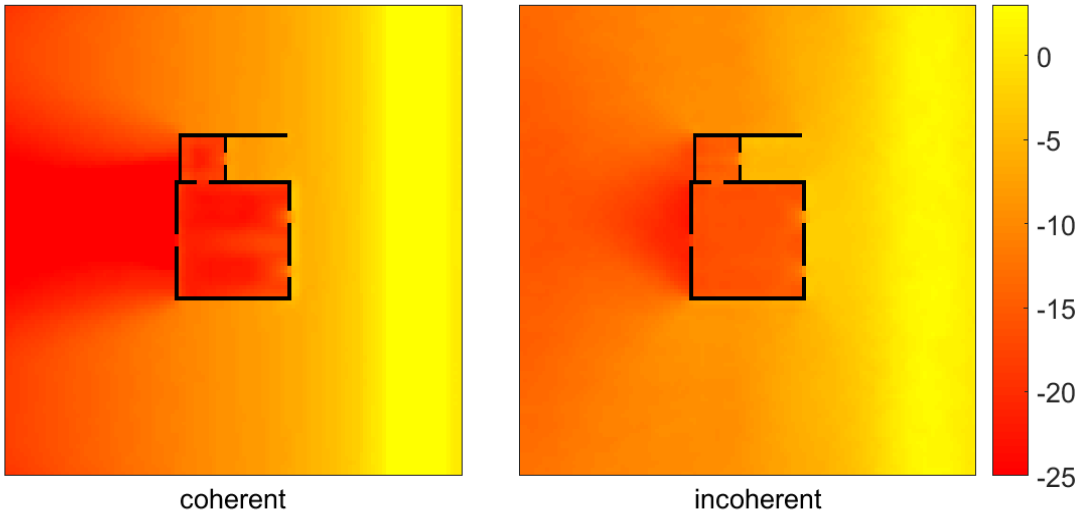

Figure 4.1: Total loudness (dB) due to coherent vs. incoherent extended source.

our experiments. Our simulation is computed on a cuboid domain with perfectly matched layer (PML) [35] absorber to suppress spurious reflections from the domain boundary.

4.3 Incoherent signal synthesis

A key component of our system is signal generation for an extended, spatio-temporally incoherent source, representing $s(x,t)$ in (3.1). Figure 4.1 shows the problems that arise from a coherent source. The scene in this experiment is a two-story beach house facing a much longer ocean surf source. When all points covered by the source emit a phase-locked Gaussian derivative pulse, an overly bright band of constructive interference forms near the source (right of image) and fringes (spatial oscillations) in the occluded area behind

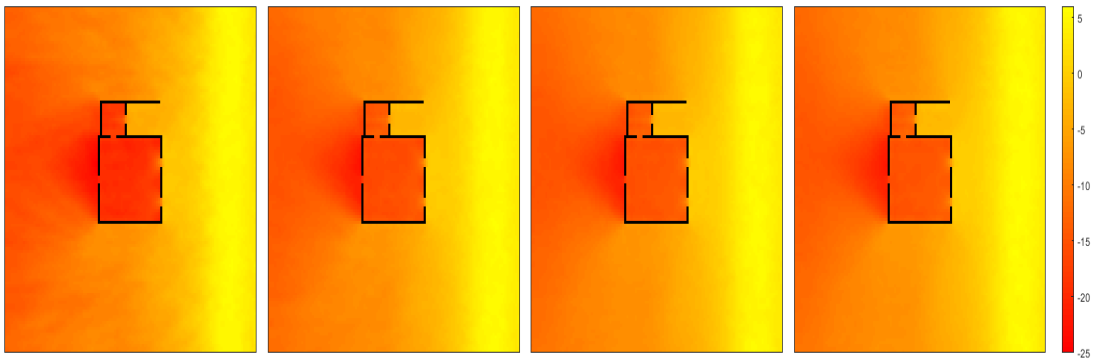

Figure 4.2: Convergence with increasing ΔN. From left to right: $\Delta N = 63, 250, 1000, 4000$, visualizing total loudness in dB scale.

the house. Inside the house, note the increased spatial variation and overall attenuation compared to the incoherent case. Our new incoherent source yields a smoother and more physically motivated total loudness field.

Coherence is the tendency of wave field observations at different times and places to be correlated. We design our source to exhibit complete spatial incoherence and as little temporal coherence as possible while satisfying the frequency content constraints discussed in the previous section.

Spatial incoherence A source much smaller than the sound wavelength can be treated as a point. When it is bigger, the signal it emits must be decorrelated over its extent as well as over time. To ensure spatial incoherence, we simply generate independent signals in each FDTD grid cell covered by the source. This approach is simple and achieves spatial incoherence up to the grid resolution; the smaller the voxels, the more closely the result approximates perfect spatial incoherence.

Temporal incoherence A sequence of independent, zero-mean random samples is fully incoherent but isn't bandlimited and so incurs numerical dissipation. To reason about bandlimiting the signal appropriately for simulation, we consider the discrete Fourier domain, where this ideal white noise exhibits constant amplitude and independently random phase at each frequency bin. The signal we desire has the same random phase and constant amplitude but only across the ΔN frequency bins in the available band $[f_{\min}, f_{\max}]$. A longer simulation increases ΔN and thus incoherence. Figure 4.2 shows how the simulation converges to a smooth result as ΔN increases.

Online synthesis The most straightforward way to generate the source signal is via Fourier synthesis: for each voxel, construct the discrete signal in Fourier space by setting the magnitude of each frequency bin to the PSD target at that frequency, and its phase as a random number, then take the inverse Fourier transform. This method precisely controls the spectrum, but must store the full temporal signal at every source voxel. For large sources, this overwhelms storage for the FDTD state itself, which only updates pressure using the field at current and prior time step.

To reduce memory, we synthesize the source signal by filtering zero-mean white noise on-the-fly during simulation. The resulting source signal is $s(t, x) = w(t, x) * h(t)$ where $*$ denotes time-domain convolution. Note that the filter $h(t)$ is independent of position; we assume elementary sources share the same spectral content.

We implement h as an order-n_h IIR filter, which accesses samples from the most recent n_h timesteps. The memory required is proportional to the product of the number of source

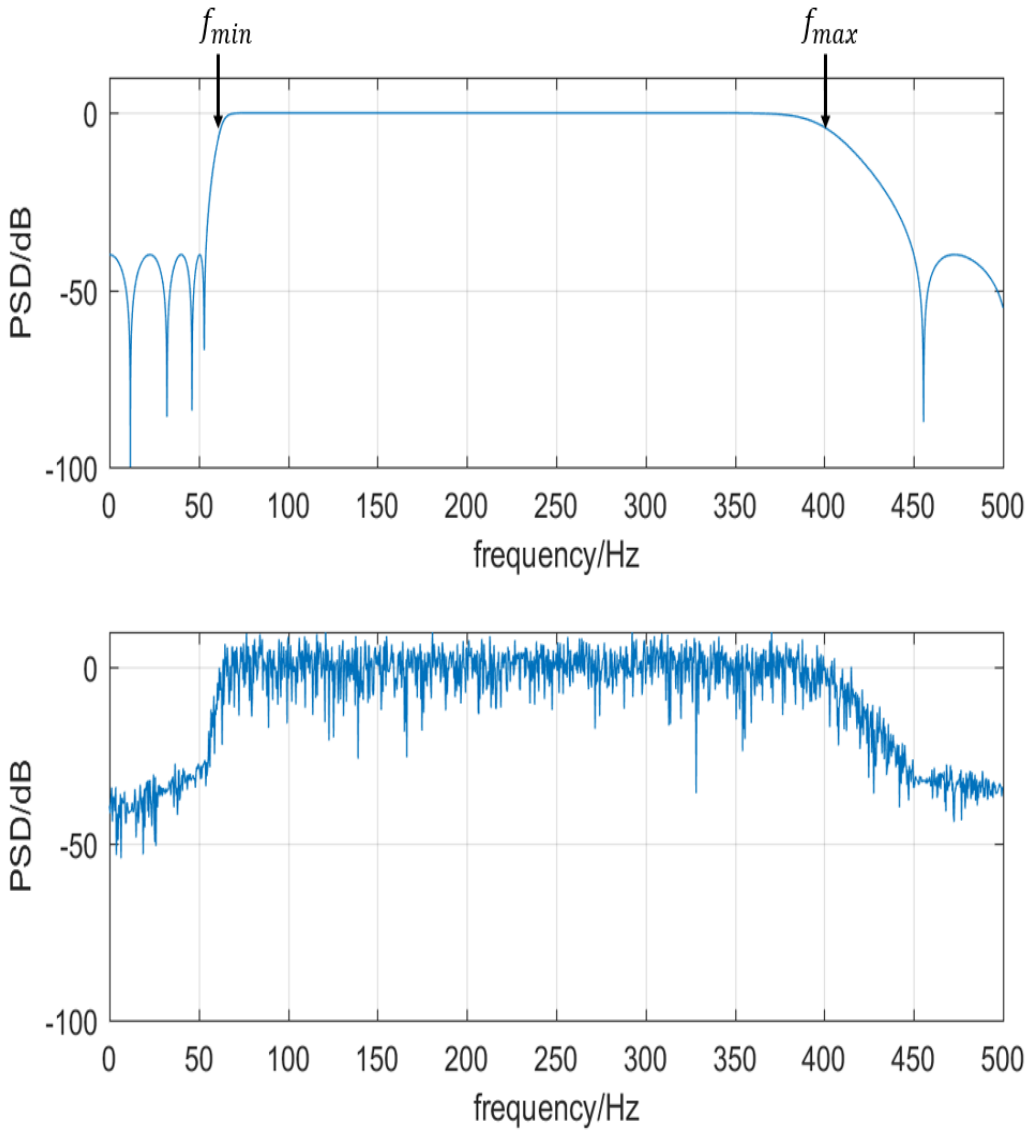

Figure 4.3: Online incoherent synthesis. Top plot is the desired magnitude response of the source signal; bottom is its PSD when actually generated via on-the-fly IIR filtering of white noise.

grid cells and the filter order n_h, independent of the simulation's duration. Figure 4.3 compares PSDs between the target and the actual stochastic signal.

Filter details For a similar rolloff factor in the transition between pass-band and stop-band, an IIR filter requires lower order compared to an FIR filter. It incurs more phase distortion, but this is not troublesome because the signal's phase is randomized.

We apply a band-pass Chebyshev type-II filter [46] and use Direct Form II for recursive filtering [45]. The pass band is set as $[62.5, 0.36\,C\,f_n]$ in Hertz, and the stop bands as $[0, 20]$ and $[0.675\,C\,f_n, f_n]$. This guarantees the pass-band is contained in the non-dissipative frequency range. The ripple (ratio between the largest and smallest magnitudes of filter's frequency response) allowed in the pass band is 6dB and the attenuation factor of the stop bands is 40dB. We use filter order $n_h = 16$. Memory demanded by online source synthesis is below one megabyte even for an extended source comprising tens of thousands of voxels.

4.4 Encoder

At each simulation voxel, our encoder computes flux at the next simulation time step and accumulates its low-order SH power distribution. The resulting parameters fields are then spatially down-sampled and compressed as in [31].

Acoustic power flux density is computed at each voxel and each time step. We recover the time-averaged directional power distribution at any spatial point x (suppressed below)

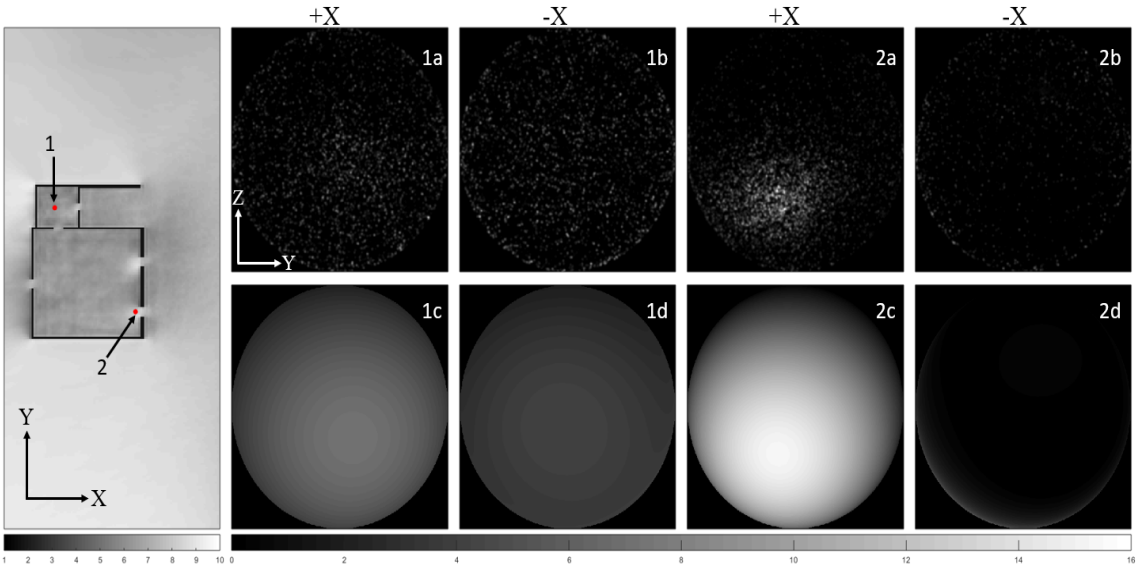

Figure 4.4: Directional analysis example. Left column: overall directionality. Right four columns: recorded directional RMS power distribution $\sqrt{E(\Theta)}$ for the two marked locations. (1a)/(1b) show the $\pm X$ hemispheres of the directional power distribution at location 1. (1c)/(1d) show the corresponding SH reconstruction. (We visualize $\sqrt{E(\Theta)}$ instead of $E(\Theta)$ to emphasize low-power spots.) (2a)-(2d) show the same analysis for location 2.

as follows. At every time step t, take the instantaneous flux to form the unit vector $\hat{f}(t) \equiv f(t)/\|f(t)\|$ and associate the instantaneous power $p^2(t)$ to that single direction, followed by time-averaging, yielding

$$E(\Theta) = \frac{1}{T} \int_0^T p^2(t)\, \delta(\Theta - \hat{f}(t))\, dt \tag{4.3}$$

where Θ represents a direction and $\delta(\Theta)$ is the Dirac delta function in direction space.

SH projection We encode directional power distribution as a smooth spherical function by projecting to order-n SH via

$$E_{l,m} = \frac{1}{T} \int_0^T p^2(t)\, Y_{l,m}(\hat{f}(t))\, dt. \tag{4.4}$$

where $Y_{l,m}$ are the n^2 real SH basis functions [42]. We use $n = 4$. We note that the integral in (4.4) can be evaluated progressively without storing history for $p(t)$ and $f(t)$. Our streaming encoder accumulates into the n^2 SH coefficients $\{E_{l,m}\}$ at each time step. A smooth reconstruction of input power distribution is then given by

$$E(\Theta) \approx \sum_{l=0}^{n-1} \sum_{m=-l}^{m=l} E_{l,m}\, Y_{l,m}(\Theta). \tag{4.5}$$

Windowing To avoid directional ringing [44], we filter the $\{E_{l,m}\}$ through the Kaiser window [28]

$$K(l) = \frac{I_0\left(\beta\, \sqrt{1\, -\, (l/n)^2}\right)}{I_0(\beta)}. \tag{4.6}$$

This is an efficient approximation of the theoretical window (DPSS) maximizing main lobe energy. Here, I_0 is the zeroth-order modified Bessel function of the first kind and β is a positive parameter adjusting the shape of the window. We set $\beta = 5$.

The final SH coefficient becomes

$$E'_{l,m} = K(l)\, E_{l,m}. \tag{4.7}$$

We assume windowing and drop the prime in the following.

Normalization The DC component, $E_{0,0}$, corresponds to total power received at the listener. A scale factor is applied to all coefficients $\{E_{l,m}\}$, such that the maximum loudness (see below) over the iso-surface 1m away from the source is normalized to 0dB.

Example Figure 4.4 visualizes RMS power distribution $\sqrt{E(\Theta)}$ and its SH reconstruction at two listener locations in BeachHouse. A simple metric for overall directionality used in the figure's left column is the ratio between the L2 norm of the linear SH coefficients and the DC component. The figure also shows the power distribution for two listener positions. Location 1 lies inside the small reverberant room and is more directionally diffuse compared with location 2 near the portal facing the extended source. Our compact SH representation is able to capture such strong anisotropy introduced by portals and corners.

Compression As with other energetic acoustic parameters [31], our encoded SH coefficients $\{E_{l,m}(x)\}$ form a spatially smooth field. We apply a similar pipeline of spatial smoothing, quantization and compression.

We encode the DC component in logarithmic space via total loudness $L = 10 \log_{10} E_{0,0}$, and the higher-order SH coefficients $l > 0$ in linear space relative to DC via $E_{l,m}/E_{0,0}$. Representing total loudness in decibels accords with human perception; encoding relative SH coefficients retains directional information even in highly occluded cases. L is clamped within the range $[-60, 6]$dB. The rest of the coefficients are bounded by the ratio of the max value of the SH basis function to DC (i.e. by $|Y_{l,m}(0,0)/Y_{0,0}|$) for non-

negative spherical functions; $[-2, 2]$ encompasses the range of values we've encountered for windowed functions with $n = 4$.

The parameters are then spatially down-sampled using simple averaging over a 1m×1m×1m cube centered at x. Only parameters in voxels unoccupied by scene geometry and visible to x are included. Parameters are quantized using the quantum 1dB for L, which is the just-noticeable-difference for loudness [16], and 0.04 for the others. Finally, the parameter fields are compressed along each x scanline (as with PNG images) and (lossless) LZW applied to the running difference. This pipeline attains a compression factor of over a million with respect to the raw parameter fields, yielding an output of about a megabyte per ambient source in a scene. The wave simulation data, whose size is on the order of terabytes, does not need to be stored by our streaming encoder but is processed at each time-step and discarded.

4.5 Runtime

Runtime decoding is similar to [31]. Each of the n^2 decoded parameters is trilinearly interpolated over the visible voxels of the surrounding cube's 8 vertices to the current listener position, yielding the propagated directional power distribution $E(\Theta)$.

Spatialization The human auditory system relies on interaural phase (IPD) and loudness difference (ILD) cues for localizing sounds in direction space. The HRTF (head-related transfer function) captures the mutual phase shift and shadowing introduced by the hu-

man head and shoulders at the two ears, for incoming coherent wavefronts over various directions. For each direction, it tabulates the complex transfer function for both ears at a dense (˜200) set of frequency bins. Binaural rendering is performed by taking a source's mono-aural emitted signal and convolving it with the HRTF via complex multiplies at each frequency bin. This is computationally costly, but necessary for point sources as they radiate spatially coherent wavefronts with a salient IPD.

For chaotic and extended sources, we observe that phases of the arriving field in different directions tend to be mutually uncorrelated, making IPD cues less detectable. We thus render only the frequency-dependent head shadowing effect (ILD) given the runtime listener head pose and the incoming spherical power distribution, $E(\Theta)$.

In the frequency domain, denote the (complex-valued) HRTF as $H(\Theta, f)$ where Θ is sound arrival direction and f is frequency. We set the gain of the sound signal at frequency f as

$$g(f) = \sqrt{\int_{\Omega} E\left(\mathcal{R}^{-1}(\Theta)\right) \left\|H(\Theta, f)\right\|^2 \, d\Theta}, \tag{4.8}$$

where Ω is the direction space, E is the reconstructed directional power distribution from (4.5), and $\mathcal{R}$ transforms directions from the head to the world coordinate system.

We divide the audible frequency range into n_H sub-bands. For the i-th band $[f_0^i, f_1^i]$, the respective gain is

$$g^i = \sqrt{\int_{\Omega} E\left(\mathcal{R}^{-1}(\Theta)\right) \, H^i(\Theta) \, d\Theta}, \tag{4.9}$$

where $H^i(\Theta)$ is the average HRTF power over the sub-band

$$H^i(\Theta) = \frac{1}{f_1^i - f_0^i} \int_{f_0^i}^{f_1^i} \left\|H(\Theta, f)\right\|^2 \, df. \tag{4.10}$$

Representing H^i using the same low-order SH approximation used for the propagated power distribution E, the spherical integral in (4.9) becomes a simple dot product of a pair of length n^2 vectors, followed by a square root. Finally, the resulting n_H scalar gains are applied to the representative clip, which is separated into n_H sub-bands using online equalization filters. Our implementation uses the built-in equalizer from the XAPOFX library.

We use $n_H = 4$ sub-bands having center frequencies f_c at 125, 600, 2400, and 9600Hz, with two-octave bandwidth, $[f_c/2, 2\,f_c]$. The SH projection of the HRTF is done as a precomputation (see below) and does not change at runtime.

SH rotation We currently support azimuthal rotation of the listener; general rotation is a simple extension [18]. If the listener's head is at azimuthal angle θ, the block-diagonal SH rotation matrix is

$$M(\theta) = \text{diag}\{M_0(\theta);\ M_1(\theta);\ \ldots;\ M_{n-1}(\theta)\}, \tag{4.11}$$

where $M_k(\theta)$ is the $(2k+1) \times (2k+1)$ matrix

$$(M_k)_{ij}(\theta) = \begin{cases} \cos((k+1-i)\,\theta), & \text{if } i = j, \\ \sin((i-k-1)\,\theta), & \text{if } i + j = 2k+2, \\ 0, & \text{otherwise,} \end{cases} \tag{4.12}$$

for $i, j = 1, 2, \ldots, 2k+1$. This matrix transforms the SH vector E from the world to the head coordinate frame, where it is dotted with each of the n_H HRTF vectors to yield the gains.

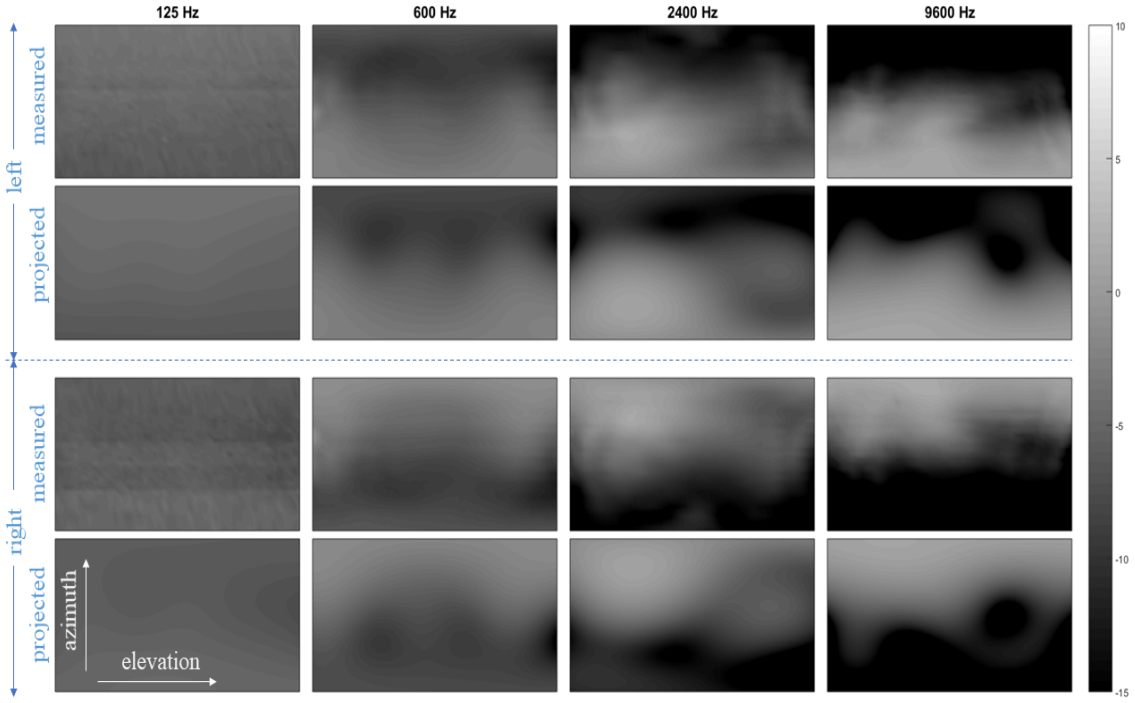

Figure 4.5: HRTF directional power representation. We used subject #3 from the CIPIC database. First (left channel) and third (right channel) rows show average spectral power over each sub-band; second and fourth rows show the corresponding least-squares reconstruction using low-order SH. All fields are visualized in dB. The horizontal and vertical axes represent elevation (-45 to 235 degrees) and azimuthal angle (-80 to 80 degrees) using the interaural coordinate system [1]. As expected, low-order SH reconstructs a smooth approximation of the measured data.

HRTF projection We use the public domain CIPIC database [1]. After converting HRIR measurements to frequency domain HRTFs $H(\Theta, f)$ via the discrete Fourier transform, we project the average HRTF power in each sub-band to SH via least-squares optimal projection [43]. Note that measurements in the database have sampling gaps around the poles.

Figure 4.5 shows our representation based on average HRTF power distribution $H^i(\Theta)$ and its corresponding SH reconstruction.

4.6 Results

Precomputed simulation and streaming encoding are performed on a single desktop with Intel i7 CPU @ 3.70GHz and 32G RAM. The technique is integrated with Unreal Engine 4™. Precomputation data for our scenes is summarized in Table 4.1. Bake times vary proportionally to scene volume and only very weakly with the number of source voxels (compare the two sources in ZenGarden). We note some manual domain adjustment was performed for the two sources in Titanpass so their two domain sizes are not identical.

Runtime memory cost is around 1 MB per extended source and decoding cost about 100 microseconds per frame. HRTF-based rendering is also lightweight (transformation of the SH vector E from world to head space plus the dot product between the length-n^2 HRTF vector and the rotated SH vector E in each of the n_H sub-bands), making our framework immediately practical.

We demonstrate four scenes in the supplementary video (link: https://vimeo.com/292495561): BeachHouse, Outpost23, TitanPass and ZenGarden. BeachHouse includes a single source in the form of a long cuboid representing the beach; the single source in Outpost23 comprises three large industrial fans. Two separate sources are included in TitanPass (waterfall and stream) and ZenGarden (rain falling into a water pool and rain falling everywhere

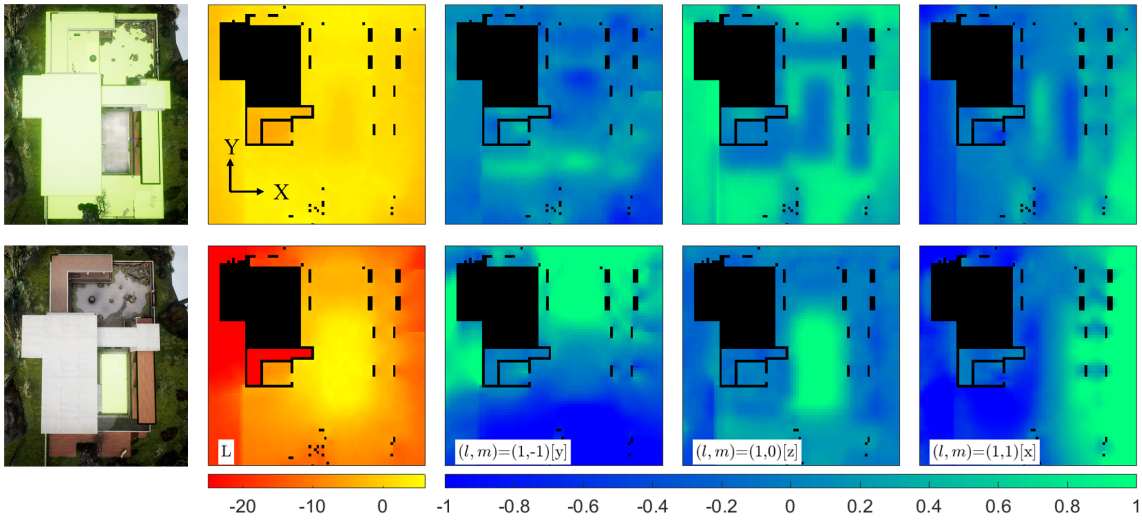

Figure 4.6: Parameter fields in ZENGARDEN. We precompute the time-integrated directional energy arriving at every potential listener in a scene using real spherical harmonics (SH). Visual rendering is on left with extended sound source marked bright green; right panels show horizontal slices of our encoded 3D parameter fields. The figure's two rows show results for two ambient sources (marked in green) with separately encoded propagation effects: rain hitting the ground and rooftops (top) and rain hitting a rectangular water pool (bottom). Second column shows the total loudness field in dB. Remaining columns show first-order SH coefficient fields. We encode up to third order.

else), allowing two different representative signals in each scene. The rain source is generated by tracing drops vertically from the ceiling of the simulation domain and placing an incoherent point where it first intersects scene geometry.

Figures 4.6 and 4.7 show parameter fields for ZENGARDEN and BEACHHOUSE, respectively. They are smooth indoors and outdoors in scenes with complex portals and occluders. High-frequency spatial oscillation due to interference is avoided. Overall, our system is well spatialized, providing smooth loudness and directional cues from environmental propagation.

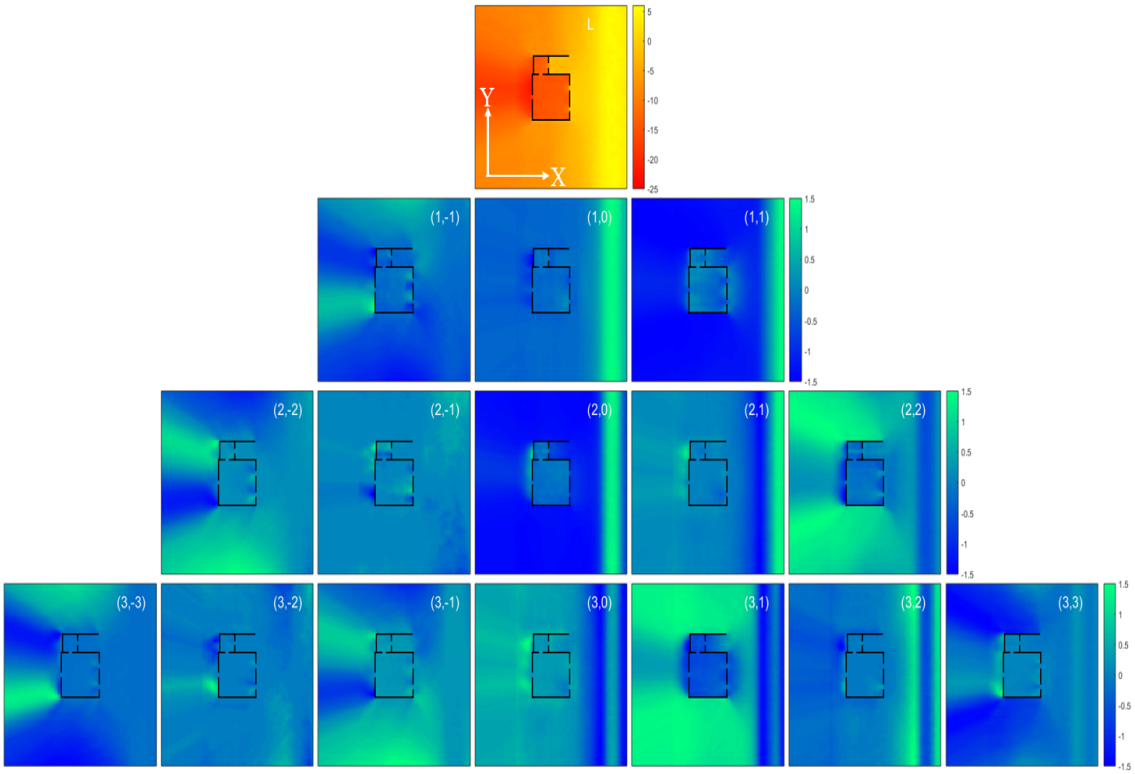

Figure 4.7: Parameter fields in BeachHouse. Top row: total loudness field L in dB. Bottom rows: higher-order ($l > 0$) SH coefficients relative to DC, $E_{l,m}/E_{0,0}$, before windowing. Total loudness field L captures the spatial variation of ambient sound loudness and the high-order ($l > 0$) SH coefficients $E_{l,m}/E_{0,0}$ encode the directional power. All encoded parameter fields are spatially smooth.

BeachHouse This simple scene demonstrates shadowing and guiding of sound around the building and through its doors and windows. Inside the house, directionality indicates positions of the nearby portals. Outside, the ocean gets louder. Loudness variation is smooth and stays steady as the listener moves along the beach. Next to and behind the building, shadowing substantially reduces loudness while increased directionality can be heard at the sides and back corners of the building.

OUTPOST23 Loudness varies smoothly even when walking directly in front of the fans. Our directional rendering clearly points the way towards these sources as the listener navigates.

TITANPASS The waterfall is a big, loud source that dominates nearby. Directionality is clearly audible in the cavern by the falls. Descending past the lower falls, the burbling stream becomes the main sound. It is highly directional as heard from the bank and gets louder and more surrounding (isotropic) in the enclosed channel.

ZENGARDEN A softer rainfall sound from a large area source covering the ground and rooftops permeates the scene, joined by splashing sounds of rain falling into the fish pond. Directionality towards the pool is clearly rendered while the background rainfall remains audible. Moving along the walkway next to the pool, rain sounds are shadowed smoothly by pillars supporting the roof.

Incoherent vs. coherent comparison The video also compares the coherent point source technique of [31] with our incoherent extended source method in the BEACHHOUSE scene. We placed nine point sources evenly along the coast where each emits a (coherent) Gaussian derivative pulse simultaneously in a single precomputed simulation. Because the nine sources are precomputed, runtime cost is similar to our technique; the cost would be significantly higher than ours if the sources were independently controlled at runtime. Several artifacts can be observed with this alternative. Moving along the ocean source, loudness wobbles unnaturally due to interference among the point sources. The shadow

Table 4.1: Precomputation data

scene/(source)	# scene voxels	# source voxels	scene surface area (m^2)	time steps	dimensions (m)	bake RAM (GB)	bake time (h)	encoded (MB)
BeachHouse	2.5×10^6	17.0×10^3	0.5×10^3	9.2×10^3	$45 \times 80 \times 8$	2.2	2.0	0.68
Outpost23	1.4×10^6	3.6×10^3	32.8×10^3	9.1×10^3	$40 \times 40 \times 10$	1.4	15.0	2.1
TitanPass (waterfall)	2.0×10^6	1.7×10^3	9.7×10^3	9.2×10^3	$20 \times 60 \times 21$	2.0	6.9	1.0
TitanPass (stream)	3.5×10^6	1.9×10^3	9.3×10^3	9.2×10^3	$20 \times 80 \times 28$	3.0	12.3	1.6
ZenGarden (rain-ground)	2.4×10^6	46×10^3	14×10^3	9.1×10^3	$50 \times 70 \times 8$	2.4	14.1	1.6
ZenGarden (rain-water)	2.4×10^6	4.0×10^3	14×10^3	9.1×10^3	$50 \times 70 \times 8$	2.4	13.8	1.6

behind the house is also unrealistically sharp. Good results for large sources require numerous point sources (thousands in our experiments) with uncorrelated phase.

4.7 Conclusion

Assuming ideal incoherence of an ambient source over both time and spatial extent, we make it practical to render its propagated effects through a complex 3D scene. Unlike ge-

ometric methods in acoustics and CG, our method computes an Eulerian PDE simulation. Wave effects like diffraction are included and cost remains insensitive to scene complexity and source size. We show how the incoherent source signal can be evaluated efficiently and propose a streaming encoder to capture the time-averaged directional power distribution of the propagated response in terms of low-order spherical harmonics for each listener position. The resulting 3D parameter fields are smooth and compressible, needing just a few megabytes per source. Our system inexpensively generates convincing ambient effects that give salient information about the scene.

Many limitations remain to be addressed in future work. Frequency-dependent propagation effects are a straightforward extension which require extracting the relevant power in frequency bands from the response at each listener position and then performing the same analysis we propose for each band. More challenging extensions break our assumption of ideal incoherence, to capture near-field effects when sound events are individually audible, or add parameters that depend on the transient response for partially incoherent sources (e.g. delay/directionality for outdoor echoes).

Our directional rendering method can be improved. Its rationale is that with incoherent ambient sources, phases at the two ears are uncorrelated and only frequency-dependent shadowing effects are noticeable. We thus apply the same representative signal equalized at the two ears according to the listener's head shadowing effects, with matching phase. Decorrelating these phases [50] is more natural and would probably increase the feeling of envelopment.

Finally, Eulerian simulation could be applied to conventional light rendering to exploit

its computational independence on scene complexity and source size. Since we don't expect it will be competitive to simulate big scenes at visible light wavelengths, such simulation will need to mitigate diffraction effects at longer wavelengths and augment the wave simulation with BRDF/scattering models for more coarsely discretized geometry.

CHAPTER 5

ACOUSTIC TEXTURE RENDERING FOR EXTENDED SOURCES IN COMPLEX SCENES

5.1 Introduction

Natural ambient sound sources can be modeled as the superposition of spectrally-similar atomic *source events* overlapping in time and distributed over the source's spatial extent, such as individual rain drops, oscillating bubbles in a stream, or bird tweets in a flock. We perceive these elementary events in aggregate. Yet not all detail is lost and we are able to hear certain statistical properties which we call the *acoustic texture*.

The goal of this chapter is to capture and render the *acoustic texture* variation from ambient sound sources in virtual environments. The problem is challenging. Brute force rendering where each individual source event's emitted signal is convolved with its acoustic impulse response captures all audible detail but at tremendous CPU cost. Chapter 4 and [53] model variation in overall *loudness* and *directionality* from a single wave simulation with sustained noisy source signals (section 4.3), producing a constant far-field texture lacking the spatial variation we wish to model. To extract *acoustic texture* information from precomputed simulation, the simulation source signal discussed in section 5.2 is necessary, which stays closer to reality by stochastically emitting band-limited pulses over time and source extent. We then perform a wave solution to efficiently propagate a massive superposition of sound radiated from the source events. At each potential listener

location, we deconvolve away the injected pulse to obtain an *aggregate impulse response* capturing times and loudnesses of *arrival events* after propagation through the scene. We employ a sparsity-regularized deconvolution in the time domain that contends with numerical dispersion errors to produce sharp estimates of event arrival time.

Our key contribution is to formulate and compactly encode acoustic texture with a novel *event density function* (EDF) computable from this propagated aggregate response. Assuming a stationary stochastic process, we observe that it is the distribution of event loudnesses and their arrival rate at the listener that determine the perceived acoustic texture. We thus define the EDF as the temporal frequency, or density, of events received at the listener as a function of their loudness.

The EDF distills effects of distance and environmental interaction on sound as it propagates from a given extended source volume to the listener location. With the listener near a large source, a broader, flatter EDF is obtained representing a variety of event loudnesses with some nearby loud events heard over a background of quieter ones. Further away or in a more occluded part of the scene, the EDF shifts to be quieter overall and becomes more peaked. This reflects extensive mixing of similarly quiet sounds, resulting in a faint and noise-like texture which gives a distant impression. Temporal density is increased by reverberation, resulting in an EDF with larger integral (total event frequency) indoors compared to the same sound source in free field with no scene geometry. Figure 5.1 provides more detail on the relationship between acoustic texture and EDF.

The EDF varies smoothly over space, yielding a compact representation after compression in our test scenes (about 1MB per source including an orthogonal directional

representation). Run-time rendering decompresses and interpolates the EDF at the dynamic listener location, and then performs granular synthesis to superimpose many fragments of sound, or *grains*, with statistics governed by the decoded EDF. The grains may be generated procedurally or extracted from recordings. Our results and accompanying video show that extracting a field of EDFs driven by sound propagation in a scene yields a natural-sounding perceived texture. Overall, we propose the first practical system to capture real-time spatial variations in acoustic texture for extended sources in complex scenes.

5.2 Precomputed sound transport

Similar to the paradigm in chapter 4, the first phase of our system is a precomputation that depends only on the scene and the source volume, in which we simulate sound wave propagation. The results of this simulation will be processed to derive the event density function (EDF) at listener positions throughout the scene for later use in run-time rendering.

Our precomputation still uses the finite difference time domain (FDTD) method [47] to propagate sounds. Unlike the noise-like source signals used in chapter 4, in this work, the simulator models the ambient source as an aggregation of events and will be explained later.

Care must be taken to keep the simulation stable and avoid numerical dispersion and dissipation errors. Our experiments use a voxel size of $\Delta x = 0.15$m and a time step of

$\Delta t = 0.25$ms, which implies a Nyquist frequency of 2000Hz and a Courant number of $1/\sqrt{3}$.

Source event signal The source event pulse introduced into the wave solver should follow two rules of thumb to minimize numerical error. First, its power spectral density should be bandlimited in the frequency domain, vanishing near DC (0Hz), to avoid residual particle velocity, and before the simulation Nyquist, to reduce dispersion errors that cause ringing in FDTD. Second, it should be compact in the time domain, minimizing overlap among neighboring event signals so that arriving events can be distinguished.

The Gaussian derivative is a compact, zero-mean signal. We tune its single parameter to avoid extensive energy approaching the simulation Nyquist via

$$s_0(t) = -\frac{t\sqrt{e}}{\sigma} e^{-\frac{t^2}{2\sigma^2}} \tag{5.1}$$

where $\sigma = 3\,\Delta t$ and the maximum amplitude of $s_0(t)$ is 1. We also define the width of the pulse t_s such that $|s_0(t_s/2)| = 0.01$, yielding $t_s = 5.4$ms. Figure 5.2 shows the signal and its power spectral density. We normalize the source pulses introduced into the solver by scaling $s_0(t)$ such that the expected sound power emitted by a 1m^3 source volume is 20dB.

Source event placement We then introduce this source pulse identically at random onset times over the source duration T_S and random 3D locations over the source. Since the arrival event density at the listener integrates over the source, we fix the temporal event density across the entire volume of the source as $d_0 = 0.1/t_s\ s^{-1}$. A simple tradeoff governs our choice of this density. Higher density allows a shorter, less expensive simulation but

makes it more likely that arrivals will be incorrectly merged. Lower density avoids mergers but requires a longer and more expensive simulation to collect sufficient statistics. While $T_S = 2$s works well in our tests, for larger sources it can be increased to ensure EDF convergence. This is readily ascertained by analyzing the smoothness of spatial variation in EDF statistics, as shown in Figure 5.6.

5.3 EDF extraction

Wave simulation produces a time-varying pressure response $p(t; x)$ at each listener position. This data is processed as the simulation runs to accumulate a measurement of the EDF at each x. Doing this robustly and efficiently is challenging and must contend with numerical error. Dispersion errors in a bandlimited FDTD simulation cause "ringing", creating many lagging and attenuated copies of a single arrival event. Standard frequency-domain deconvolution exacerbates ringing and impractically requires storing the entire response in memory. We seek a streaming method which manages event aliasing and avoids assembling the entire response before extraction.

5.3.1 Deconvolution

Deconvolving the pressure response $p(t; x)$ with respect to the source pulse $s_0(t)$ recovers the aggregate impulse response $h(t; x)$ by inverting

$$p(t;\, x) \;=\; h(t;\, x) \;*\; s_0(t). \tag{5.2}$$

We propose a sparsity-regularized time-domain deconvolution which reduces sensitivity to ringing. Interpreting (5.2) as a sparse superposition of time-shifted copies of $s_0(t)$, deconvolution becomes an L_2 minimization problem with L_1 regularization (LASSO [48]):

$$\arg\min_{h} \frac{1}{2}\left\|A\,h - p\right\|_2^2 + \lambda\,\|h\|_1 \tag{5.3}$$

where each column of the square matrix A is a time-shifted version of the elementary pulse $s_0(t)$ and λ is a regularization parameter.

Applying this idea directly on the entire space-time pressure response is expensive. We instead extract $h(t; x)$ in non-overlapping segments of length $T_0 = 10\Delta t$ and accumulate EDF statistics from each. Since the input pulse has a finite duration t_s, one must consider an input window with $t_s/2$ extra duration on either side to ensure that any output peak is fully captured within the analyzed input segment of the pressure response. Thus, $p(t)$ is analyzed using overlapping input segments of duration $t_s + T_0$ that increment by T_0. In each segment, the matrix A is $n \times n$ where $n = (t_s + T_0)/\Delta t$; the first column contains the shifted pulse $s_0(t + t_s/2)$ and the last column contains $s_0(t - T_0 - t_s/2)$. Note that in this setting, the shift step size of source signal $s_0(t)$ equals simulation time step Δt exactly. However, there is no fundamental limit stopping us from further decreasing the shift step size, yielding a super-resolution version of deconvoluted response (down to subpixel level) though it increases expense.

We use the alternating direction method of multipliers (ADMM) to solve (5.3) in each segment [7]. We regularize using $\lambda = 0.1\,\|A^T b\|_\infty$; a standard choice that balances sparsity and convergence rate. The result is an estimate of $h(t;\, x)$ of length n. We discard the overlapping portions of the time segment from this output, using only the middle portion

(of width T_0) for EDF accumulation.

Figure 5.3 shows the output of this method on input representing our wave simulator's pressure response when emitting a few pulses into the free field (i.e. without scene geometry). While the input is degraded by numerical dispersion, deconvolution still recovers narrow spikes.

5.3.2 Accumulation and encoding

At each simulation voxel, deconvolution over each time segment yields a corresponding segment of the aggregate impulse response from which it is straightforward to accumulate the EDF. We apply peak detection (using a simple relative min/max detector on three values adjacent in time), extract the peak's amplitude A, compute its loudness via $L = 10 \log_{10} A^2$, and accumulate peak loudnesses into a running histogram. Because max loudness is unknown when encoding begins, we accumulate into a histogram of conservatively large span: $[-60, 60]$dB with bin size of 3dB. The overall computation allows streaming, requiring only the time-varying pressure response over one time segment and the accumulated histogram (40 bins) as its stored state.

Once simulation completes, the histogram is converted to the encoded EDF as follows. We first extract the maximum loudness received quantized to 3dB, L_m, as a separate channel. We then store temporal densities at the next 12 loudness bins of width 3dB descending from L_m. The event density within each bin is divided by the total temporal density over the whole simulated source, denoted d_0. Encoding relative rather than absolute density

factors out dependence on the (somewhat arbitrary) density of the simulation source and allows run-time substitution of a source of different density via simple scaling. The relative densities are quantized in the range $[0, 20]$ with a quantum of $1/3$. These values are empirically determined.

Overall the process produces 13 parameter fields containing, for each listener location, L_m and relative event densities in 12 EDF loudness bins offset from it.

As with previous work [30] [53], the parameter fields are spatially smooth and can be compressed. We compress the raw data by applying lossless LZW algorithm to the running difference along x scanlines.

5.3.3 Spatialization

In addition to the above EDF information, we also extract and encode an overall directional distribution of energy at each listener position x as the simulation runs. The method follows chapter 4 and directional energy is aggregated in terms of spherical harmonics(SH). We then aggregate directional energy in terms of spherical harmonics (SH). At run-time, the mono sound signal is synthesized as discussed in the next section, and then spatialized per this spherical energy distribution (while ignoring inter-aural phase) to produce an output binaural signal.

5.4 Run-time rendering

Run-time rendering performs granular synthesis [36], which controls meso-scale sound texture by mixing micro-scale acoustical grains. The key aspect of our work is to control synthesis so it corresponds to an extended source shape in a physical scene. We assume some procedure to generate randomized grain sounds such as rain drops or bird tweets, discussed later. Our goal is to render how these user-provided grains would sound at the listener if they were randomly emitted from the source and propagated in the scene. Our approach is to randomly draw grains and superpose them while obeying the acoustic texture as captured by the EDF.

5.4.1 Grain blending

We first decompress and spatially interpolate at the current listener location x, to obtain the EDF $E(L; x)$ where L is loudness. We henceforth drop the listener position, x. Recall that $E(L)$ relates the relative temporal event density to loudness. We employ a straightforward method that is cache-friendly and obtains real-time performance. It scales and adds grain signals to the output audio buffer such that their loudness and temporal density statistics respect $E(L)$.

Because grains can be longer than the audio buffer size (in our case 1024 audio samples), our method queues active grains. For each, it stores the grain onset time, the amplitude, and any other parameters needed to finish synthesizing its signal coherently

across multiple output audio buffers. Two simple algorithms drive synthesis, detailed in Figure 5.4. Note the difference between this grain blending idea and conventional convolution operation between dry audio and impulse responses, as the grains blended are de-correlated.

We define overall grain density (i.e., grains generated per second) as $D = d_0' \int E(L)\, dL$. The integral yields the total events per second at the listener relative to the source's event density. This is multiplied with d_0', the absolute event density of the source, which is specified by the sound designer based on physical or artistic concerns and can be modified in real-time if desired. This yields D, the expected number of grains we will superpose per second for the current audio buffer.

To stochastically draw the grain amplitude requires a simple procedure we perform on the fly. The computation is minimal because $E(L)$ is represented with just 12 loudness bins. We first compute $E(L)/D$ to form a probability density function (PDF) providing the probability for each loudness bin. We then integrate to form the cumulative distribution function (CDF). Generating a uniform random number q in $[0, 1]$, we compute the loudness corresponding to q by inverting the CDF. Finally, we relate grain amplitude to loudness via $A = 10^{L/20}$.

As the listener moves, the EDF continually changes. The above process accommodates such modification by stochastically drawing from the current EDF at each audio sample. If the listener stands still, the rendered grain statistics converge to the (static) distribution over the next few audio output buffers.

Atmospheric attenuation Without atmospheric attenuation, high frequencies can sound unnaturally harsh when standing far from the sound source. Our solver lacks such modeling, but we find a simple approximation sufficient in practice. We employ standard analytical formulae relating propagation distance to atmospheric attenuation [15]. We conservatively estimate distance to the source as $r \approx 10^{(L_0-L)/20}$, where L_0 is the empirically determined loudness of the source at 1m distance, and L is the EDF's average loudness across all arriving events at the listener location. Our formula assumes attenuation from a point source; since the loudness falls off more gradually for an extended source, we underestimate the distance. We implement the computed frequency-dependent attenuation using a 4-band equalization filterbank with center frequencies of {125,600,2400,9600}Hz.

5.4.2 Grain synthesis

Our rendering method requires a collection of grains as input; they should be spectrally and semantically similar and short. Most of our grains persist for just a few tens of milliseconds, though our longest (human utterances) are several seconds. We then randomly select one grain from the collection each time a grain is added in the algorithm from Fig. 5.4. The grain synthesis parameter P becomes a simple index into the collection. Our examples use several different approaches to generate a grain collection.

Water sounds (both drop impacts and flows) apply a procedural model based on bubble oscillations [12]. Though it's fast enough to perform purely procedural synthesis in real time, we still pre-generate a random collection of 1000 grains, thereafter treating them as

recordings to simplify the implementation. An individual bubble sound has profile

$$g(t) = a\ \sin(2\pi f_0 t)\, e^{-d\,t} \tag{5.4}$$

where f_0 is the oscillating frequency, damping rate

$$d\ =\ \frac{1}{0.043 f_0\ +\ 0.0014 f_0^{3/2}} \tag{5.5}$$

and amplitude a proportional to $f_0^{-3/2}$. See [12] for more details. The oscillating frequency f_0 of the pre-generated bubbles are sampled within range $[f_{min},\ f_{max}]$ with its probability density function (PDF) proportional to $1/\sqrt{f_0}$ [11]. For water stream sound, $[f_{min},\ f_{max}]$ = [500, 8000] Hz. For rain-pool sound, $[f_{min},\ f_{max}]$ = [2500, 20000] Hz.

We manually segmented recorded clips into individual events to obtain the crowd and bird-flock grain collections. We obtained 45 grains (tweets) from a single recorded clip (starling flock), and 2000 grains (English utterances) from a database of recordings [41]. For the lapping waves, we segmented a recording applying a Kaiser-Bessel sliding window [6], with 2 second width, 1 second sliding step, and $\beta = 10$, yielding a collection of 19 grains.

5.5 Results

We precompute a single FDTD simulation and encode the EDF and SH directional parameter fields for each scene per sound source, which takes 5 – 10 hours on a desktop

computer. In all our test scenes, the compressed data is about 1MB per sound source. Our run-time implementation is integrated in Unreal Engine 4.

Results for ambient soundscapes in four scenes are included in the supplementary video (link: https://youtu.be/SjdLBB6HOdk). RiverHouse includes a linear stream of water with nearby building geometry. RainOverPool models the sound of light rain hitting a rectangular water pool in a house and garden scene. StylizedKingdom demonstrates outdoor acoustic texture effects from a flock of starlings in a tree canopy and water lapping around a lake. BabblingCrowd includes a single extended source in a small room representing a babbling crowd.

In most scenes, run-time grain density, d_0', varies from 1-5× what was encoded. The exception is procedural water sounds, where we apply a larger scale factor (around 100× for rainfall and 500× for stream flow). Absolute source density at run-time can be computed from these factors by multiplying by the global factor $d_0 = 18\text{s}^{-1}$. Recall that arrival density at the listener is further modified by the EDF. As the listener moves, we plot the EDF corresponding to that location, annotated with green bars for the 50th (median) and 95th percentile loudnesses across all arriving events, and the average loudness. The video also demonstrates the enhanced realism provided by rendering the spatially-varying acoustic texture using the EDF, compared to a fixed one (labeled "loudness variation only" in the video) as in [53].

Our approach first produces a mono signal blending across all grains, and then spatializes the blended result using an aggregate directional representation collected over the entire source. Ideally, each grain would be spatialized separately since each exhibits a sep-

arate arrival direction. Our simple model compactly encodes the main effects, becoming less convincing with a collection of directional and highly recognizable/distinguishable grains, as with a listener standing in the midst of a babbling crowd.

RIVERHOUSE. Near the stream source, crisp gurgling sounds are audible. The acoustic texture stays nearly constant as the listener walks along its bank. As the listener moves away from the stream, these distinct gurglings gradually disappear and the texture smoothly changes to become more noise-like. Inside the house, reverberation causes a Gaussian EDF shape, essentially capturing the room's decay time characteristic. Behind the house outdoors, edge diffraction rather than reverberation serves to mix events arriving around the sides of the building from across the source, resulting in a unimodal but non-Gaussian EDF without loud outliers and conveying a distant source. Outdoors but between two walls, the result sounds like the expected fusion of indoors and outdoors.

RAINOVERPOOL. Individual rain drops are audible close to the pool, becoming gradually indistinct farther away, and more suddenly noise-like and denser as the listener enters the house.

STYLIZEDKINGDOM. The flock of starlings tweeting spatializes well overhead. The scene also demonstrates our system's ability to render multiple sources.

BABBLINGCROWD. A babbling crowd gathers in a small room in this scene. The listener begins in a large indoor courtyard outside the room's open door. Acoustic texture varies convincingly as the listener moves from near to far or side to side across the opening. Inside the room in the midst of the crowd, we obtain a plausible result even though grain utterances fail to merge into recognizable conversation.

5.6 Conclusion

Our system models extended sound sources by propagating identical source event pulses distributed randomly over the source through a synthetic scene via a single wave simulation. We show that loudness variation alone fails to capture the resulting nuanced spatial variation. We instead formulate the event density function to represent how acoustic texture varies in 3D scenes, and show how it can be extracted from simulation and used to govern run-time synthesis. The encoded parameters are spatially smooth and compress to a very compact representation. Our system obtains convincing variation of acoustic texture in complex scenes and allows real-time movement of the listener.

Acoustic texture also varies with direction, which we currently don't model. Standing between an extended source and the door to a room, the source sounds clearer in the ear facing the source and more indistinct in the ear facing the door through which room reverberation gets mixed. Our method also neglects to capture frequency-filtering effects in arriving events, such as from edge diffraction, though this information is available from the simulation. Finally, a complete solution would offer some way to capture near-field

effects where our model breaks down and the correlation of onset time and energy of successive events becomes audible. As shown in our crowd babble scene, while a reasonable rendering is obtained as long as utterances remain unintelligible, the full cocktail party effect, preserving both the directional and semantic content of sounds made by each nearby speaker, remains for future work.

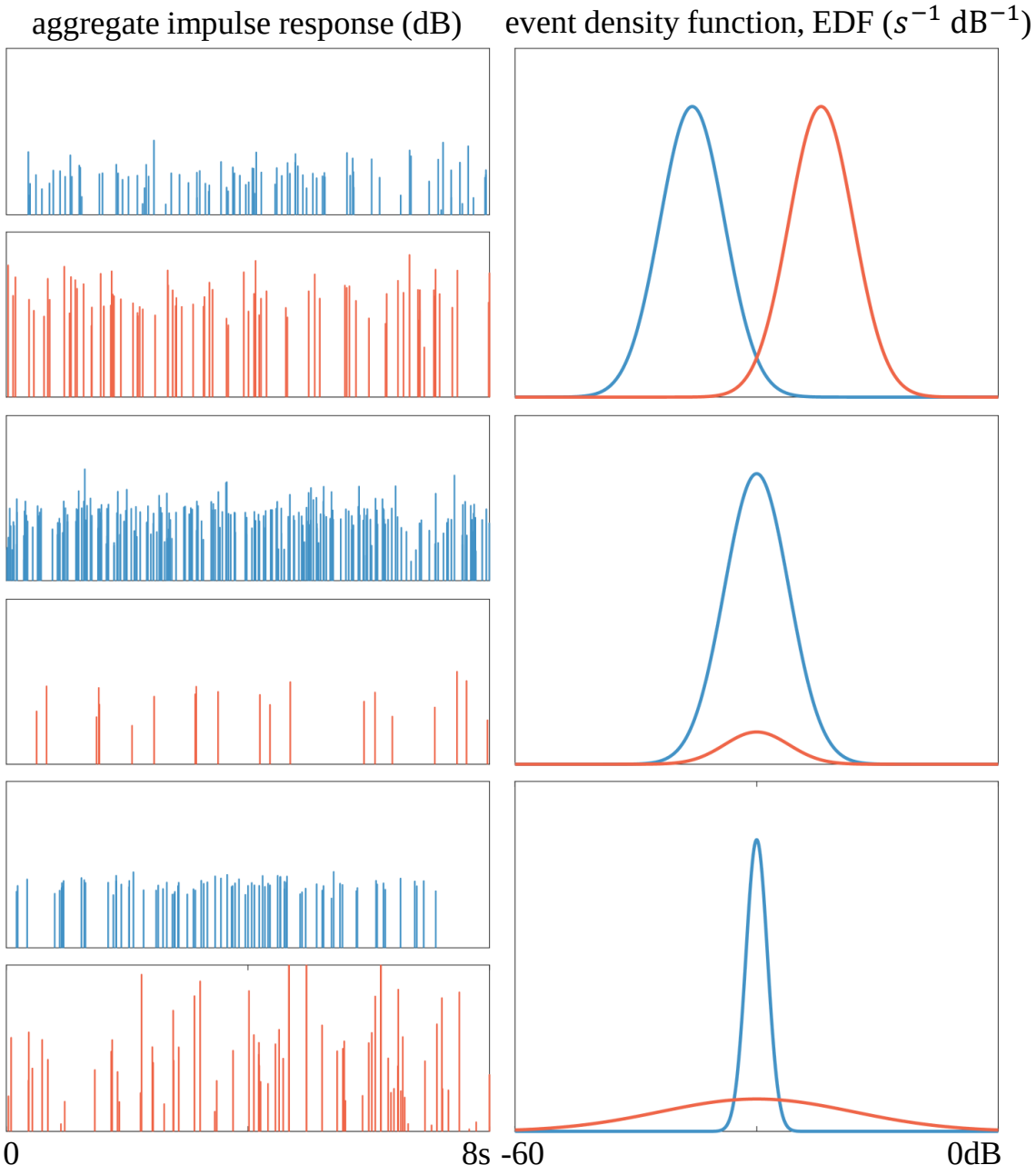

Figure 5.1: Three examples relating the aggregate impulse response of arrival events, left, and the corresponding event distribution function (EDF), right. Refer to the accompanying video to hear the corresponding audio renderings. Shifting the distribution horizontally, along the loudness axis, corresponds to making all arrivals louder or quieter (top). Scaling up the EDF without changing its shape corresponds to adding more events drawn from the same loudness distribution (middle). Making the EDF broader, with a fixed area below the curve, changes the arrivals from a train of similar loudnesses to a train with large variations in loudness (bottom).

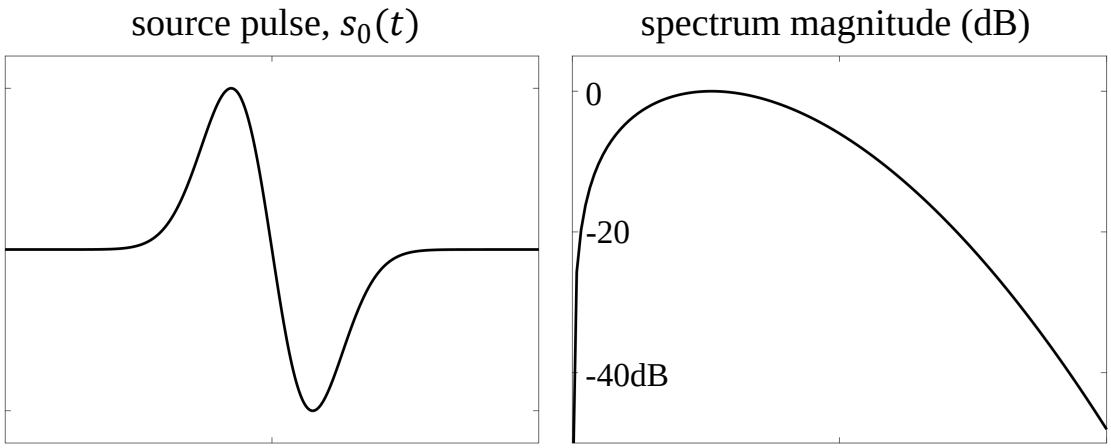

Figure 5.2: Band-limited source event pulse used by the solver, $s_0(t)$. Left: Source pulse in time-domain. Right: energy spectral density.

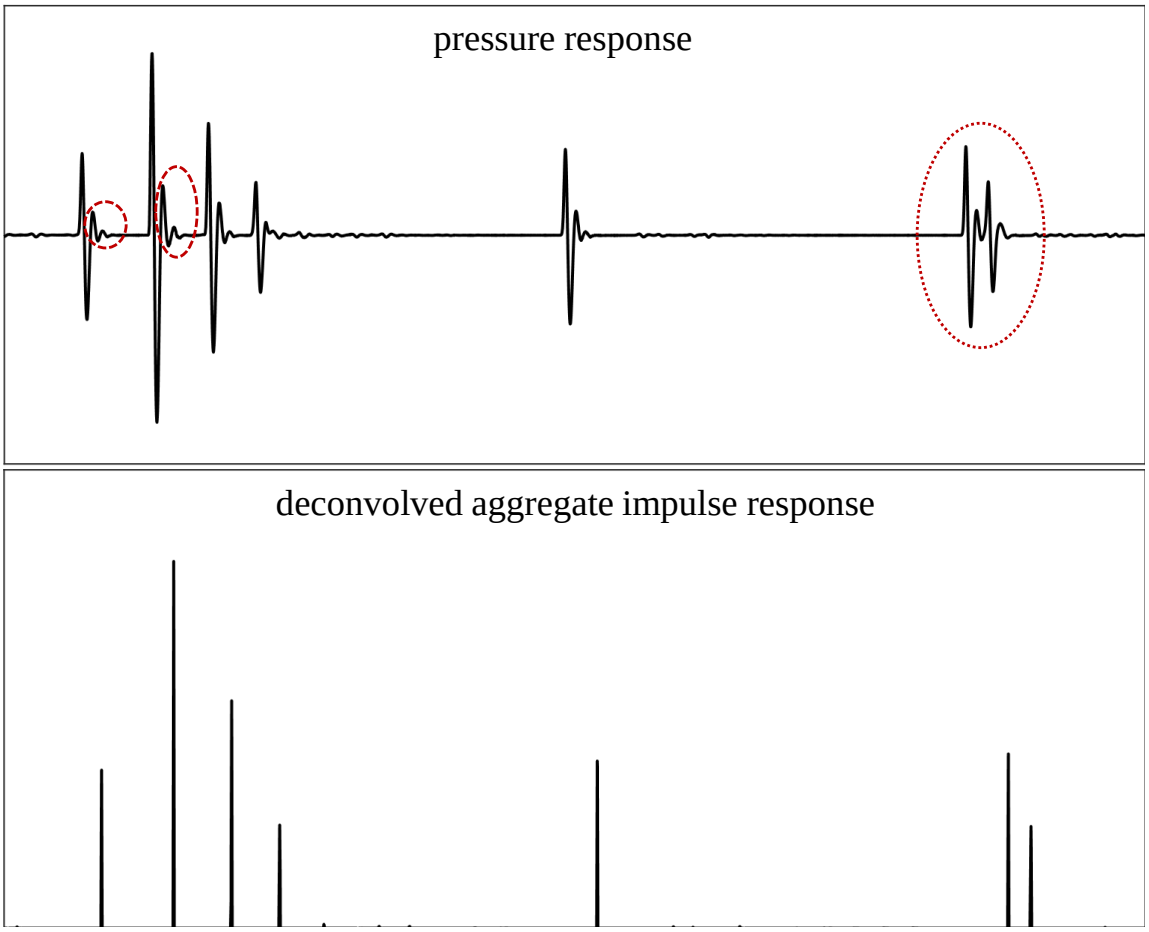

Figure 5.3: L_1-regularized least-squares deconvolution. The top image shows the received pressure signal; the bottom shows the deconvolved result. Our method is designed to cope with the ringing resulting from numerical dispersion in order to best distinguish arrival events. Parts of the pressure response marked by the dashed red areas show ringing "copies" of the original Gaussian derivative pulse injected into the simulation. Our method does not confuse these for additional arrivals. The dotted red area shows where two pulses overlap; our method correctly resolves two separate arrival events.

R: audio sample rate [44100Hz]
rand(): uniform random number generator in [0,1]

Runtime synthesis:

for each output audio buffer
Call **Add new grains**
for each active grain
  synthesize signal remainder and add to buffer, scaled by A
  if grain now complete, dequeue

Add new grains:

for each time sample t in buffer
  if rand() $< D/R$, //decide to generate grain
    draw grain amplitude A from EDF E
    generate other grain synthesis parameters P
    enqueue grain at (t, A, P)

Figure 5.4: Granular rendering of the event density function (EDF).

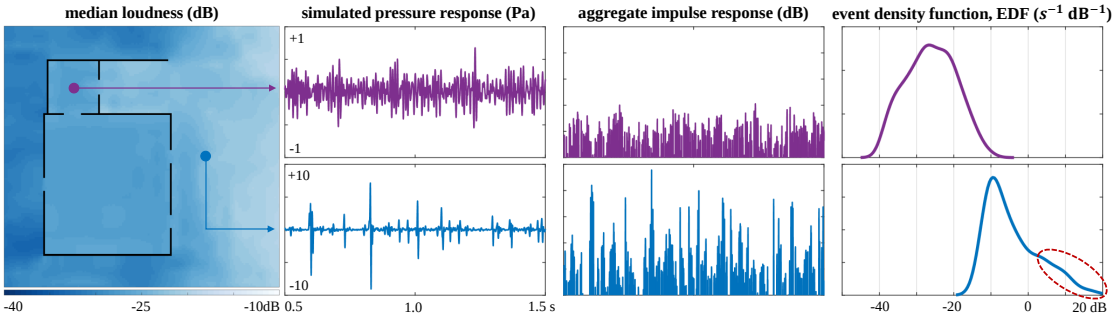

Figure 5.5: Capturing acoustic texture variation in RiverHouse. The source in this simple scene represents a river located to the right of the house. We illustrate what our method does at two particular points: inside a small room (top row) and outside nearby the river (bottom row). Running a numerical simulation of sound transport which stochastically emits pulses over the source into this scene, we obtain pressure responses for each listener position (second column). After sparsity-regularized deconvolution, these are converted to an aggregate impulse response representing arrival events at the listener (third column). We then extract an event density function, encoding temporal density of arrivals at the listener position as a function of loudness (fourth column). Source events are multiplied inside the house due to reverberation yielding many similarly quiet arrivals. Whereas outside and near the source, fewer arrivals are recorded overall, with some infrequent but loud events that stand out from the rest (dashed red area).

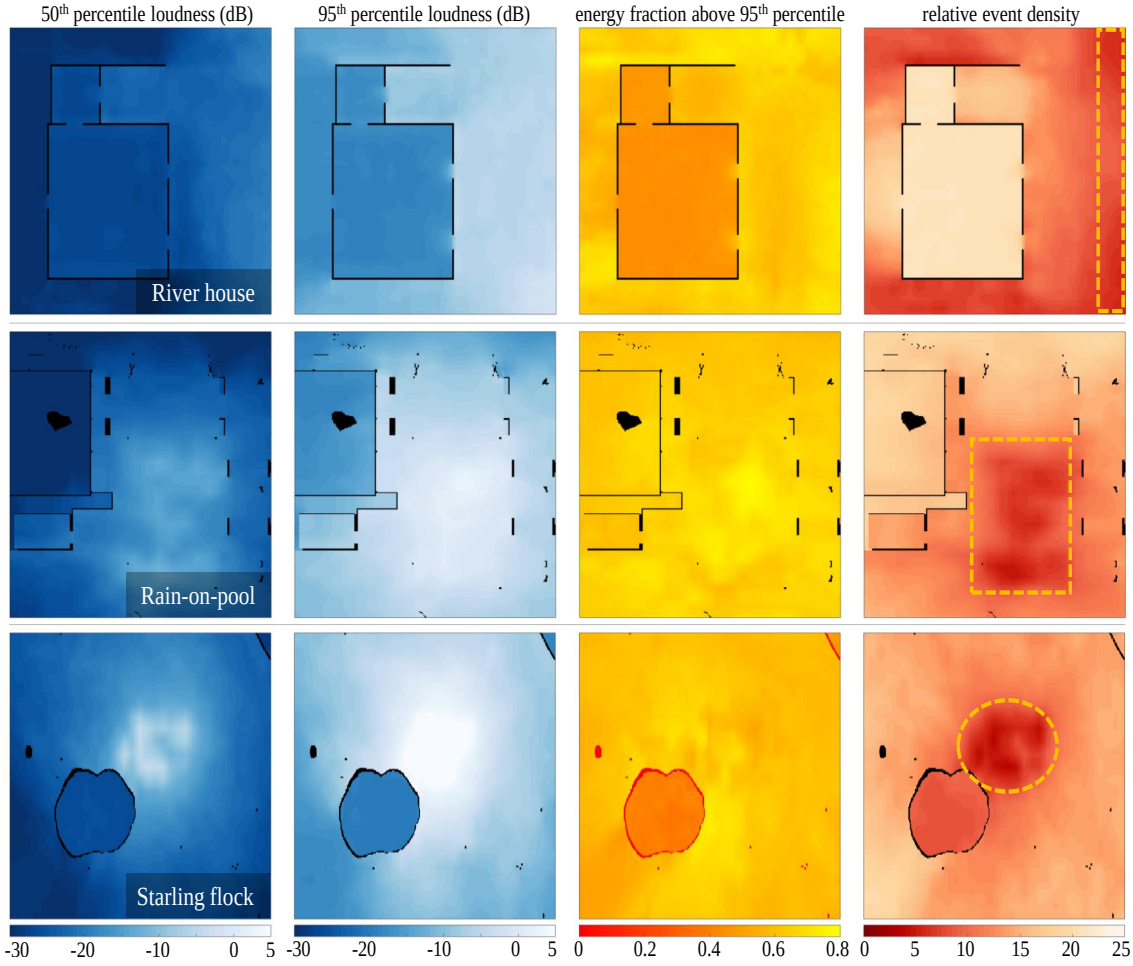

Figure 5.6: Spatial maps of EDF characteristics. We show results for three scenes each with a single ambient source, indicated by the dashed yellow area in the fourth column. The leftmost two columns show median (50th percentile) and 95th percentile loudness over all arriving events, respectively. These indicate the smoothness of our parameterization. The third column plots the fraction of energy remaining above the 95th percentile: brighter areas indicate more loud arrivals that stand out over the rest. Note how this property diminishes inside the river house, conveying the increased mixing of grains that happens indoors due to reverberation. The fourth column maps overall arrival density (over all loudnesses). Note how this property increases in enclosed spaces from reverberation. Darkening of event density right at the source location is due to the omission of (non-salient) quiet arrivals whose loudness falls below the 12 loudness bins we encode.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this dissertation, we introduced the problem of computing spatial sound in a virtual environment and focused on a specific type of sound: ambient sound. We discussed the difficulties of directly using previous point sound source methods for ambient sound simulation and rendering. Ambient sound sources have extended source volume in 3D space: extending previous geometric acoustics methods leads to slow convergence and it can not simulate sound propagation at interactive rates; extending prior physical wave acoustics methods for point sources leads to unphysical interference artifacts and unnatural shadowing effects.

We therefore proposed a new method that precomputes an Eulerian simulation of physical sound wave propagation with static ambient sound source, which is spatio-temporally incoherent. The incoherent source signals can be synthesized efficiently in simulation. The time-averaged directional power distribution of received sound per listener location is captured by a streaming encoder in terms of low-order spherical harmonics. At run-time, we propose a lightweight binaural ambient sound rendering technique that leverages the decorrelation of sound phases among arrival directions and generates convincing ambience consistent with the scene.

We then observed the ambient sound texture is spatially varying and is influenced by

scenes as well. We call it *acoustic texture*. To statistically describe acoustic texture, we first proposed the notion of event density function (EDF). EDF represents the arrival rate and the amplitude distribution of the elementary sound events propagated from an ambient source towards a listener. To simulate and encode EDF per listener position, we modify the ambient source signal synthesis method in wave solver and capture the EDF distribution per voxel in simulation, along with directional spherical harmonics coefficients. At run-time, granular synthesis with an overlap-add scheme is used for acoustic texture reconstruction and synthesized sound texture is spatialized using previous technique.

6.2 Future work

The story of ambient sound in virtual worlds is not over yet, and there exist limitations that remain to be addressed in future work.

Texture synthesis improvement Chapter 5 uses granular synthesis method to construct acoustic texture at run-time. However, this requires a collection of high quality grains and could be challenging. To mitigate this, one potential solution takes advantage of texture synthesis works [23, 24, 3]. The user only inputs a single near-field ambient recording sound clip and a series of perceptually similar (but independent) sound clips can be produced by texture synthesizers subsequently. At run-time, the acoustic texture is approximated as a superposition of these sound clips and the blending can be guided by our acoustic simulation framework.

Directional acoustic texture In the physical world, acoustic texture is dependent on arrival direction. Beside a water stream, the sound coming from stream directly exhibits more dominant transient details compared with sound coming from other directions due to geometry reflection. At this point, our framework computes acoustic texture as a function of 3D positions only and is unaware of texture variation over 2D arrival direction space.

Towards simulating a cocktail party The ultimate goal of spatial computing of ambient sound is to simulate a cocktail party: the intelligibility of individual speech is much more enhanced inside the talking crowd compared with standing far away from party participants. How to combine the simulation strategies for ambient and point sound sources and reconstruct the cocktail party effects in 3D space is a fascinating problem.

BIBLIOGRAPHY

[1] V Ralph Algazi, Richard O Duda, Dennis M Thompson, and Carlos Avendano. The CIPIC HRTF database. In *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*, pages 99–102. IEEE, 2001.

[2] Jont B Allen and David A Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, 1979.

[3] Joseph M Antognini, Matt Hoffman, and Ron J Weiss. Audio texture synthesis with random neural networks: Improving diversity and quality. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3587–3591. IEEE, 2019.

[4] Maurice Anthony Biot and Ivan Tolstoy. Formulation of wave propagation in infinite media by normal coordinates with an application to diffraction. *The Journal of the Acoustical Society of America*, 29(3):381–391, 1957.

[5] Jens Blauert. *Spatial hearing: the psychophysics of human sound localization*. MIT press, 1997.

[6] Marina Bosi and Richard E Goldberg. *Introduction to digital audio coding and standards*, volume 721. Springer Science & Business Media, 2012.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[8] Paul T Calamia and U Peter Svensson. Fast time-domain edge-diffraction calculations for interactive acoustic simulations. *EURASIP Journal on Applied Signal Processing*, 2007(1):186–186, 2007.

[9] Chunxiao Cao, Zhong Ren, Carl Schissler, Dinesh Manocha, and Kun Zhou. Interactive sound propagation with bidirectional path tracing. *ACM Transactions on Graphics (TOG)*, 35(6):180, 2016.

[10] Richard Courant, Kurt Friedrichs, and Hans Lewy. Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische annalen*, 100(1):32–74, 1928.

[11] Grant B Deane and M Dale Stokes. Scale dependence of bubble creation mechanisms in breaking waves. *Nature*, 418(6900):839, 2002.

[12] Kees van den Doel. Physically based models for liquid sounds. *ACM Transactions on Applied Perception (TAP)*, 2(4):534–546, 2005.

[13] Dennis Gabor. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441, 1946.

[14] Brian Hamilton and Stefan Bilbao. Fdtd methods for 3-d room acoustics simulation with high-order accuracy in space and time. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 25(11):2112–2124, November 2017.

[15] Acoustics – Attenuation of sound during propagation outdoors – Part 1: Calculation of the absorption of sound by the atmosphere. Standard, International Organization for Standardization, Geneva, CH, 1993.

[16] IS ISO3382. Acoustics-measurement of room acoustic parameters, part 1: Performance spaces, ed. *B. Standards*, 2009.

[17] Doug L James, Jernej Barbič, and Dinesh K Pai. Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 987–995. ACM, 2006.

[18] Jan Kautz, John Snyder, and Peter-Pike J Sloan. Fast arbitrary brdf shading for low-frequency lighting using spherical harmonics. *Rendering Techniques*, 2(291-296):1, 2002.

[19] Konrad Kowalczyk and Maarten Van Walstijn. Room acoustics simulation using 3-d compact explicit fdtd schemes. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):34–46, 2011.

[20] Konrad Kowalczyk and Maarten van Walstijn. Modeling frequency-dependent boundaries as digital impedance filters in fdtd and k-dwm room acoustics simulations. *Journal of the Audio Engineering Society*, 56(7/8):569–583, 2008.

[21] Asbjørn Krokstad, Staffan Strom, and Svein Sørsdal. Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration*, 8(1):118–125, 1968.

[22] Mikko V. Laitinen, Tapani Pihlajamäki, Cumhur Erkut, and Ville Pulkki. Parametric Time-frequency Representation of Spatial Sound in Virtual Worlds. *ACM Trans. Appl. Percept.*, 9(2), June 2012.

[23] J. H. McDermott, A. J. Oxenham, and E. P. Simoncelli. Sound texture synthesis via filter statistics. In *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 297–300, Oct 2009.

[24] Josh H. McDermott and Eero P. Simoncelli. Sound texture perception via statistics of the auditory periphery: Evidence from sound synthesis. *Neuron*, 71(5):926 – 940, 2011.

[25] Herman Medwin. Shadowing by finite noise barriers. *The Journal of the Acoustical Society of America*, 69(4):1060–1064, 1981.

[26] Ravish Mehra, Nikunj Raghuvanshi, Lakulish Antani, Anish Chandak, Sean Curtis, and Dinesh Manocha. Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Transactions on Graphics (TOG)*, 32(2):19, 2013.

[27] Stanley J. Miklavcic, Andreas Zita, and Per Arvidsson. Computational real-time sound simulation of rain. In *7th Intl. Conference on Digital Audio Effects (DAFx)*, 2004.

[28] Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[29] Nikunj Raghuvanshi. *Interactive physically-based sound simulation*. The University of North Carolina at Chapel Hill, 2010.

[30] Nikunj Raghuvanshi and John Snyder. Parametric wave field coding for precomputed sound propagation. *ACM Trans. Graph.*, 33(4):38:1–38:11, July 2014.

[31] Nikunj Raghuvanshi and John Snyder. Parametric Wave Field Coding for Precomputed Sound Propagation. *ACM Transactions on Graphics (TOG)*, 33(4), July 2014.

[32] Nikunj Raghuvanshi and John Snyder. Parametric directional coding for precomputed sound propagation. *ACM Trans. Graph.*, 37(4):108:1–108:14, July 2018.

[33] Nikunj Raghuvanshi and John Snyder. Parametric directional coding for precomputed sound propagation. *ACM Transactions on Graphics (TOG)*, 37(4):14, August 2018.

[34] Nikunj Raghuvanshi, John Snyder, Ravish Mehra, Ming C. Lin, and Naga K. Govindaraju. Precomputed Wave Simulation for Real-Time Sound Propagation of Dynamic Sources in Complex Scenes. *ACM Transactions on Graphics (proceedings of SIGGRAPH 2010)*, 29(3), July 2010.

[35] Yotka S Rickard, Natalia K Georgieva, and Wei-Ping Huang. Application and optimization of pml abc for the 3-d wave equation in the time domain. *IEEE Transactions on Antennas and Propagation*, 51(2):286–295, 2003.

[36] Curtis Roads. *Microsound*. MIT press, 2004.

[37] Lauri Savioja and U Peter Svensson. Overview of geometrical room acoustic modeling techniques. *The Journal of the Acoustical Society of America*, 138(2):708–730, 2015.

[38] Carl Schissler, Ravish Mehra, and Dinesh Manocha. High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Transactions on Graphics (TOG)*, 33(4):39, 2014.

[39] Carl Schissler, Aaron Nicholls, and Ravish Mehra. Efficient hrtf-based spatial audio for area and volumetric sources. *IEEE transactions on visualization and computer graphics*, 22(4):1356–1366, 2016.

[40] John B Schneider and Christopher L Wagner. Fdtd dispersion revisited: Faster-than-light propagation. *IEEE Microwave and Guided Wave Letters*, 9(2):54–56, 1999.

[41] Shtooka. Project shtooka, 2010. Accessed: 2019-05-18.

[42] Peter-Pike Sloan. Efficient spherical harmonic evaluation. *Journal of Computer Graphics Techniques*, 2(2):84–90, 2013.

[43] Peter-Pike Sloan, Jesse Hall, John Hart, and John Snyder. Clustered principal components for precomputed radiance transfer. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 382–391. ACM, 2003.

[44] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527–536, July 2002.

[45] Julius Orion Smith. *Introduction to digital filters: with audio applications*, volume 2. Julius Smith, 2007.

[46] Julius Orion Smith. *Introduction to digital filters: with audio applications*, volume 2. Julius Smith, 2008.

[47] Allen Taflove and Susan C Hagness. *Computational electrodynamics: the finite-difference time-domain method*. Artech house, 2005.

[48] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[49] Lloyd Nicholas Trefethen. Finite difference and spectral methods for ordinary and partial differential equations. 1996.

[50] Vesa Valimaki, Julian D Parker, Lauri Savioja, Julius O Smith, and Jonathan S Abel. Fifty years of artificial reverberation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(5):1421–1448, 2012.

[51] Charles Verron, Mitsuko Aramaki, Richard Kronland-Martinet, and Grégory Pallone.

A 3-d immersive synthesizer for environmental sounds. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1550–1561, 2009.

[52] Kane Yee. Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on antennas and propagation*, 14(3):302–307, 1966.

[53] Zechen Zhang, Nikunj Raghuvanshi, John Snyder, and Steve Marschner. Ambient sound propagation. *ACM Trans. Graph.*, (6), 11 2018.

[54] Zechen Zhang, Nikunj Raghuvanshi, John Snyder, and Steve Marschner. Acoustic texture rendering for extended sources in complex scenes. *ACM Trans. Graph.*, (6), 11 2019.